Computer-Assisted Construction Planning

by

Jeffrey Hudson Rankin

BSc.Eng., The University of New Brunswick, 1991

MSc.Eng., The University of New Brunswick, 1993

A Thesis Submitted in Partial Fulfillment of

the Requirements for the Degree of

Doctor of Philosophy

in

The Faculty of Graduate Studies

(Department of Civil Engineering;

Construction Engineering and Management)

We accept this thesis as conforming to the required standard

[illegible]

[illegible]

[illegible]

The University of British Columbia

April, 2000

0-612-48697-4

(Signature) [illegible]

Department of Civil Engineering

The University of British Columbia
Vancouver, Canada

Date April 24, 2000

Abstract

The research presented is entitled Computer-Assisted Construction Planning (CACP) and it explores the computer-assisted development of initial comprehensive (include elements of scope, cost, time, performance, and organization) construction plans, which the user can customize to suit a current project.

A thorough investigation into the approaches taken for automated construction plan development was performed exploring the methods to capture and reuse planning information and the application of this knowledge to computer-assisted planning. The development of a common database for storage of all project information was completed through the selection and refinements of conceptual core models of several on-going model development projects. Validation of the developed model was performed while populating the developed database with actual project information. An interface for representing the hierarchical breakdown of project information was developed and a Case Based Reasoning approach was selected in light of the required functionality.

The research recognizes and follows the path being established for integrated construction management systems that rely on a standard representation of the industry's information requirements. The core information structures were adopted and their validity confirmed by examining a broad application for an integrated system (i.e., planning beyond time and cost). By examining the comprehensive aspects of construction planning within the framework of an integrated construction management system the research demonstrated the usefulness of applying sound information representation structures (i.e., part and type relationships). Through the application of prudent information systems development approaches (i.e., data, process, and interface analyses), and the use of case-based reasoning, the research has advanced the concepts of planning tools as they apply to integrated systems.

The research has employed sound information representation principles, explored the implementation of data management tools and techniques, and followed a rigorous development process. The resulting contributions to the research efforts in this area include, in addition to the data model and prototype deliverables, an integrated data model for construction management information, a prototype interface for an integrated construction management system, and a unique approach to assisted planning in support of a practical approach to planning.

Table of Contents

List of Tables

List of Figures

Acronyms

AEC – Architectural, Engineering, Construction
AI – Artificial Intelligence
BCCM – Building Construction Core Model
CAD – Computer-Aided Design
CACP – Computer-Assisted Construction Planning
CASE – Computer-Aided Software Engineering
CBR – Case-Based Reasoning
CIC – Computer Integrated Construction
CPE – Collaborative Project Environment
CSI – Construction Specifications Institute
IAI – Industry Alliance for Interoperability
IDEF0 – Integrated Definition for Function Modeling
IDEF1X – Integrated Definition for Information Modeling Extended
IFC – Industry Foundation Class
ISO – International Organization for Standardization
IT – Information Technology
MS – Microsoft
PMDM – Project Management Data Model
PMI – Project Management Institute
PMBOK – Project Management Body of Knowledge
RAD – Rapid Application Development
RIBA – Royal Institute of British Architects
STEP – Standard for the Exchange of Product Model Data
TOPS – Total Project Systems
UofF – Unit of Functionality

Acknowledgments

As is the case with many things undertaken in life, the journey is often more meaningful in the end than the goals or rewards that one is pursuing. To me, learning is one of the most fulfilling activities in life. The completion of this research has been a journey of sorts and I have learned so much that I wish to acknowledge the contributions of the many people who have facilitated this process.

Foremost, I wish to express my gratitude to my supervisor Thomas Froese, who agreed to take me on as one of his first Ph.D. students. I feel very fortunate to have had his guidance and support throughout this work. His exceptional research provided the standard to achieve, his perspectives have broadened my thinking, and his well-timed queries have kept me focused on completing this work. Lloyd Waugh has unwittingly served the role of mentor throughout my graduate studies career, not only in matters related to research and work but also in life, thanks Lloyd. One of the finest teachers that I have had the pleasure of learning from, Alan Russell puts meaning to the challenges that researchers face in the construction industry and motivates those that chose to pursue their solutions. In my opinion, these three individuals are among the elite in research and teaching performed in the field of Construction Engineering and Management in this country and throughout the world. It is my honor to have been associated with them.

I wish to thank members of my examining and research committees. They sent me off in the right direction after my initial twelve-month stay in Vancouver and their contributions have been invaluable to this work. Thank you to: Tarek Sayed, for impressing the need for a quality prototype development, Carson Woo, for his insights on information management, and Denis Russell for his perspectives on the profession. I also appreciate the input provided by the university examiners John Meech and Siegfried Stiemer, and the external examiner Iris Tommelein for their challenging and constructive reviews.

Thanks to my fellow graduate students at both the University of British Columbia and the University of New Brunswick and colleagues at the Construction Technology Centre Atlantic and the University of New Brunswick. I have been fortunate to meet a very diverse group of individuals, collaborate with some, reflect with others, and learn from all. An additional thanks to Kirby Ward for the expertise he shared in the implementation of the CBR tools and a special thanks to John Christian for the continued support he has provided me and the Construction Engineering and Management Group at the University of New Brunswick.

I also wish to acknowledge the absolute support provided to me by my mother and father and thank them for instilling in me a sense of heritage and for being so easy to please. To my brother and his family, Deedee, and all my other family and friends, thanks for asking how things were progressing without asking if I was ever going to finish.

Finally, my most special thanks goes to my wife Shelley, we agreed to take our adventure to the west-coast four years ago and have been blessed with good fortune since. My son, Keagan Hudson, was born shortly before the completion of this research and provided all the inspiration and necessary distractions required making its conclusion enjoyable. Without their unconditional support I would not have finished, so it is to them that I dedicate this work.

Chapter 1: Introduction

This thesis discusses research in the field of Computer Integrated Construction (CIC) with a focus towards contributions to the development of an integrated construction management system, the Total Project Systems (TOPS) project, as proposed by Froese (1995). In this regard, the research addresses the voluminous quantities of information required of an integrated construction management system by initially populating a project plan.

The term "total-project systems" (Froese et al. 1997a, 1997b) is used to refer to a conceptualized class of construction management computer systems that are defined by the following characteristics: comprehensiveness –they include a suite of applications that support a broad range of construction management functions, integration – all applications contribute to and draw from a shared pool of project information, and flexibility – they operate in a highly modular, open, flexible, and distributed framework, rather than in a restrictive and prescriptive manner.

The construction industry is characterized as having many players of multiple disciplines who are brought together at different stages throughout the life cycle of unique projects. The supporting management processes result in a reliance on an enormous amount of information produced by many sources at many levels of abstraction and detail. One of the primary goals of CIC is to simplify the methods for handling the information generated throughout the life cycle of a project, and thereby promoting more efficient and effective management processes. This is generally foreseen as a sharing of project information by all participants, across the various management supporting computer applications, applied throughout the construction process.

To facilitate the sharing of project information, common models of all project information (product, process, cost, organizational, resource, etc.) are required that detail, completely, all aspects of a project throughout its life cycle. These common models can only be feasible if standards are established for both their development and structure. One cornerstone of the TOPS approach is to build upon ongoing architecture, engineering, and construction (AEC) data standards efforts to develop common data models to support construction management data. Upon widespread acceptance of such models, it will then be feasible to capture and store the knowledge of projects for use on future projects. A second major element of TOPS is the development of the various application modules that make up the integrated system. While many of these are traditional project management applications such as scheduling, others are more innovative, such as the storage and subsequent reuse of construction planning knowledge. This is important to address TOPS' high data-entry requirements.

1.1. *Conceptualizing Integrated Systems*

In general, existing construction management software applications are not used to their current potential. This may in fact be partially due to the organizational changes that must occur in order to maximize the benefit of their application. It is certainly due, in part, to the unavailability of complete systems that are applicable to a broad range of construction management functions for a variety of project participants.

Figure 1.1 depicts the current and conceptual environment for construction management tools. In the top half of the figure, current applications can be described as a toolbox of project management tools that are all useful in their own specific tasks addressing multiple views of a project's information and frequently at various levels of detail (e.g., scheduling addresses time, estimating addresses cost). The toolbox analogy is fine for describing a tool in terms of its view, however, it does not account for the degree of overlap in the scope of project information that individual tools address which lead to difficulties in consolidating project information. For example, scheduling tools take primarily a time view but also address cost information as well and usually with a different structure and different level of detail than, say, an estimating tool would. Although the information used by current construction management tools overlaps, virtually all information exchange among these tools today occurs by printing the output of one tool on paper and entering the information in different forms into another tool.

In contrast, the bottom half of Figure 1.1 illustrates the conceptual elements of the TOPS approach. The concept is for integrated project management systems that rely on information structured according to a shared model base. The integrated tools provide much the same functionality as the contents of the toolbox, but with greater information consistency.

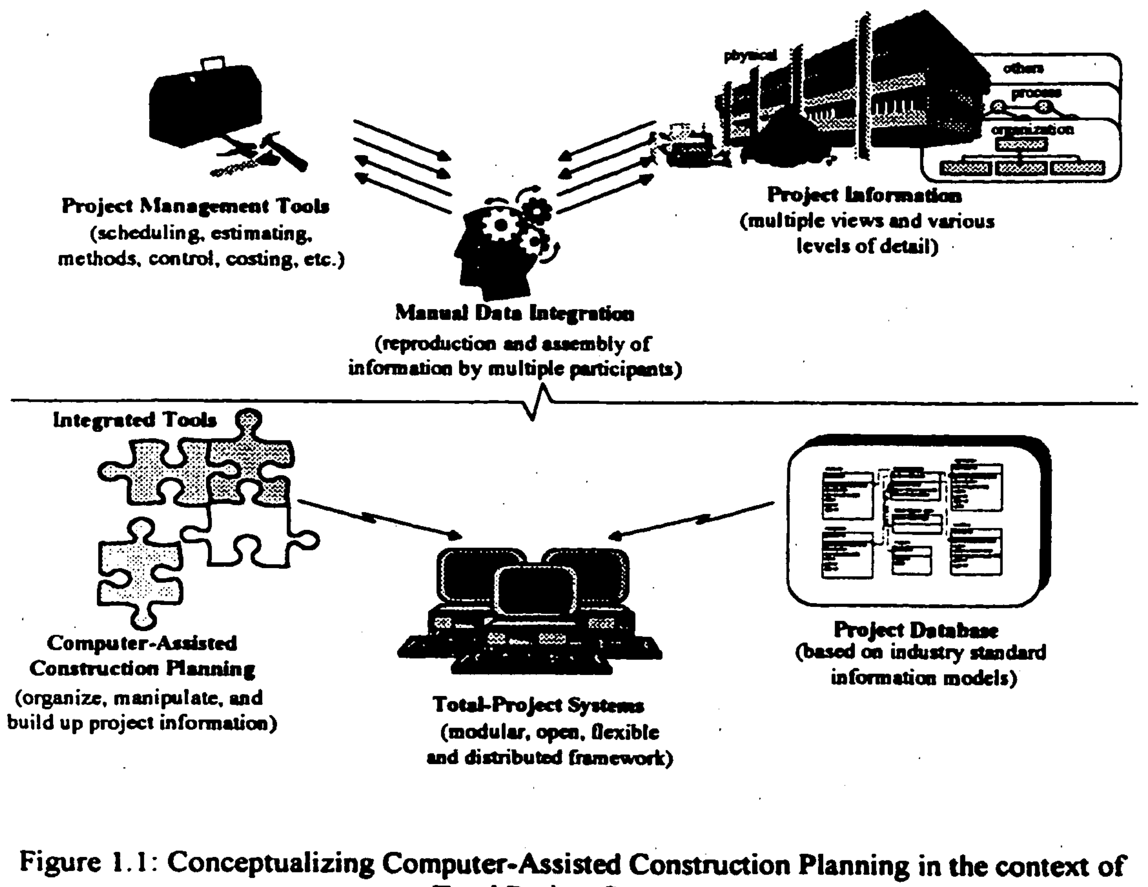

physical
others
process
organization
Project Management Tools
(scheduling, estimating, methods, control, costing, etc.)
Project Information
(multiple views and various levels of detail)
Manual Data Integration
(reproduction and assembly of information by multiple participants)
Integrated Tools
Computer-Assisted Construction Planning
(organize, manipulate, and build up project information)
Total-Project Systems
(modular, open, flexible and distributed framework)
Project Database
(based on industry standard information models)

Figure 1.1: Conceptualizing Computer-Assisted Construction Planning in the context of Total Project Systems.

Computer-Assisted Construction Planning (CACP) is one piece of the integrated project management puzzle. It supports planning in the context of integrated project management tools by storing information about construction planning, in general, and past projects, in particular, in a format consistent with a project management application information model. The scope of the planning information, therefore, includes all forms of planning such as work plans, schedules, estimates, organizational plans, document control and communication plans, etc. The result serves to initially populate the pool of project planning information that is then available for use by other specialized construction management planning tools. Therefore, it is not a new planning tool but rather an advancement of planning tool concepts in light of an integrated approach. In the context of an integrated system, applicable modifications are expected to occur from the beginning of a project's conceptual stage and could continue up to the population of information required for construction planning.

To consider the capture and later use of project knowledge, a framework is required to allow retrieval of project information at various levels of detail from a variety of views. Key components to this framework are both a practical interface and approach. With this ability to retrieve and modify templates of past project knowledge, the planning phase of project management is then partially automated. A large body of project information must be acquired, maintained, and manipulated in order for an integrated system to be useful. The computer-assisted aspect of the research requires a tool that helps to easily develop initial comprehensive plans, thereby reducing the data handling burden and increasing opportunities for the development of quality project plans. Of the research efforts investigated, no one solution addresses all the aspects that are felt necessary for a complete solution to the objectives of computer-assisted construction planning.

The concept of an integrated project management system is central to the research presented. The results of this effort come at a time when the foundations for integrated systems with the characteristics described are being established. Two essential elements of this foundation are being addressed through the development of a data model to support the planning steps of construction management and tools to manage the initial information requirements of these management processes. To further the appreciation for requirements of such a system, two simplified conceptual information usage scenarios are presented in Figures 1.2 and 1.3.

This conceptual usage scenario considers the initial entry of project information and the subsequent steps, in which multiple users (project participants) access portions of the project's information for selected management applications (e.g., scheduling, estimating, etc.). Figure 1.2 presents the usage scenario for an integrated system with a shared project database and the means to initially populate it. Each box refers to a portion of the overall project information (although they are not expected to be equal in size or complexity). The solid boxes indicate information that must be created by components within the system and the dotted boxes indicate where components use project information that was produced by other components within the system. The initial project information (shaded boxes) is built up using an application with the functionality of CACP. Then information is added to the project database by multiple users of applications that draw from this initial information and apply more detailed planning functionality (e.g., resource management algorithms, economic analysis, etc.).

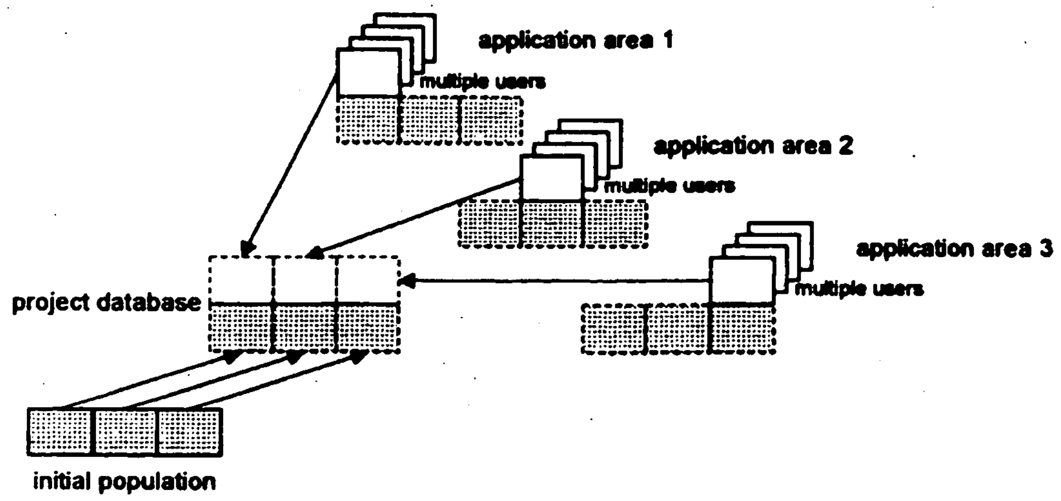


Figure 1.2: Project information with integrated systems.

In a scenario without the complete characteristics of integrated systems (Figure 1.3), the initial information requirements must be built up within each application area, and there is the potential for redundancy within each application depending on the tools applied by each project participant. Although the scenarios presented are rather basic, the ramifications should be apparent by comparing the number of boxes representing the development of project information in order to fulfill the planning processes (i.e., 19 boxes in Figure 1.2 compared to 48 boxes in Figure 1.3). Whether the information is the output of an application or simply a reproduction of existing project information, the additional steps required in the scenario presented in Figure 1.3 increase the burden of managing the project information (e.g., information consolidation from separate applications, avoiding inconsistencies among redundant data sets, etc.).

These scenarios assume that current tools will continue to be implemented in the same capacity as in today's environment, an issue that must be examined for each application in light of expected changes to industry work process and the new functionality offered in an integrated implementation. Regardless of the answers for these issues, an environment that supports integrated project management systems based on standard information models will reduce the information management burden and thus increase the management efficiency of the industry.

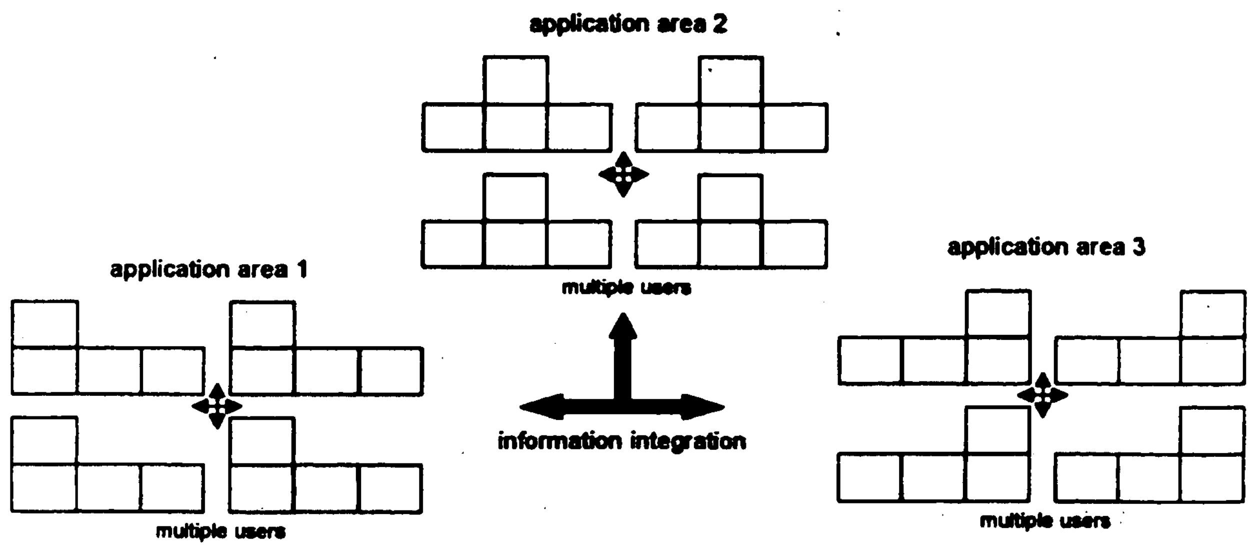


Figure 1.3: Project information without integrated systems.

1.2. Goal

The topic of this research is the *"computer-assisted development of initial comprehensive construction plans in the context of an integrated construction management system. Comprehensive suggests plans that include elements of scope, cost, time, performance, and organization."*

1.3. Objectives

The objectives of this research were as follows:

- To contribute to the development of a standard information model for the construction and related industries.
- To develop an effective and efficient interface for dealing with large quantities of information.
- To develop a hierarchical framework that facilitates the representation of planning information from separate views at different levels of abstraction.
- To demonstrate how existing planning knowledge can be represented and transferred to disseminate the benefit of shared knowledge of a construction project.
- To explore an approach to managing planning information by applying case-based reasoning to construction planning.
- To implement selected components of a system that can generate an initial construction plan for use in an integrated construction management system.

1.4. Scope

Through fulfilling the objectives listed, the deliverables of the proposed research are essentially the components of the prototype system. Deliverables of the research are as follows:

- A framework for representing a hierarchical breakdown of project information and the ability to manipulate this same information.
- A system for the assisted generation of comprehensive initial plans in support of TOPS initiatives.

Contributions to the research efforts in this area in addition to the deliverables are summarized with the following points:

- A significant effort towards the development of a project management application model.
- An example of the application of standard core models in support of integrated systems.
- An implementation of integrated systems research through the development of a prototype based on rapid application development (RAD) tools.
- Support for the future development of integrated construction management systems.

1.5. Research Methodology

As a starting point, a thorough investigation into the approaches taken for automated construction plan development was performed. This included an identification of knowledge requirements through a study of past projects and a more detailed investigation of available literature. Particular attention was paid to the manner in which historical planning knowledge is captured and reused and the methods proposed for the application of this knowledge to computer-assisted planning.

The following step was the selection and refinements of conceptual core models, and development of models to the expected level of detail required for the overall research. This included an analysis and comparison of several on-going model development projects and a collaborative effort in the development of an information model in support of project management with an emphasis towards the supporting structure for planning information. This step required a complete usage scenario for CACP, detailing the system's functionality, as part of an overall system development plan complete with a system quality plan.

A contributing effort towards both the CACP and TOPS projects was the development of a common database for storage of all project information. The focus was on the framework for how planning objects within each framework is organized in a hierarchical breakdown, how structuring of the hierarchy enables mapping between objects, and how this mapping is represented and controlled. Validation of the developed model was performed while populating the developed database with information from an actual building construction project.

Having structured and developed the database the next system development step was to fully develop the interface for representing the hierarchical breakdown of project (planning) information. This included the exploration of how such an interface can be used to manipulate and modify project information.

With the conceptual core model, database, and information interface established the next step in the research was developing the functionality required of the system for generating plans with the characteristics discussed. This began by finalizing the breadth of a plan's development, the approach to plan development, and the structure of the libraries of planning knowledge. In consideration of these approaches the knowledge software was then selected for completion of the prototype system. However, the selection of the knowledge software was from the perspective of "proof of concept," rather than the system capabilities, as ease of development was weighted high.

Having determined the most promising knowledge software methodology, the final step in the system development was to complete the integration of the database, the interface, and the knowledge application. The last stage in the research was to test the system with the project information collected previously to evaluate both the system and the complete CACP approach.

1.6. Reader's Guide

These chapter summaries are provided as a synopsis of the contents of each of the chapters to follow.

Chapter 2: Points of Departure

This chapter summarizes previous research work upon which CACP draws from and builds on. The categories of research work include: integrated construction information systems in general, specific components of integrated information systems, and approaches to automating and assisting the construction management function of planning.

Chapter 3: Project Management Data Model

The steps involved in the development of the Project Management Data Model (PMDM) are described in this chapter. The model, initially based on an existing information model is modified, extended, and simplified to fulfill the scope required for CACP.

Chapter 4: Computer-Assisted Construction Planning

The approach adopted for CACP is described in this chapter. The approach description includes CACP's unique viewpoint from an integrated project management system's perspective and describes the manner in which case-based reasoning has been adopted.

Chapter 5: Interface and Implementation

The structure of the "proof of concept" application is described in detail, through a presentation of the module interaction from a software perspective.

Chapter 6: Validation and Testing

Several examples are presented to describe CACP in action and demonstrate its use. This is accomplished with the use of multiple screen captures of the application working on sample project data gathered from an actual construction project.

Chapter 7: Contributions and Conclusions

This chapter presents the contributions to the current body of knowledge in the research of integrated construction management systems and the construction industry, specifically in the field of construction information management and construction planning. The final chapter points out that continued application of CACP and additional prototype systems to current construction projects attempting an integration of team project information is paramount to the understanding of how these future practical tools will benefit the industry.

Chapter 2: Points of Departure

Today's construction planning is essentially comprised of scheduling, cost estimating, and technology selection. These processes involve a significant management effort and are all supported, at various levels of success, by project management software. Although supporting applications are available, their total acceptance industry wide and use in an integrated approach has not yet been practically achieved. This chapter describes five areas in which research has been performed or is ongoing to support the integration and acceptance of currently available applications and future construction management supporting applications. Any criticisms offered in the following discussion are not intended to detract from the research performed. Instead, the aim is to point out possibilities for improvements to existing research and opportunities for future contributions, while indicating in a broad sense the similarities and differences in the approaches adopted for the CACP research project. Specific differences between each approach and this research are explained where relevant in subsequent chapters. Sections 2.6, 2.7, and 2.8 explore the direction of three concepts related to the development of CACP's prototype application: case based reasoning, current project management applications and their interfaces. The summary statements provided describe the points of departure for this research.

2.1. Computer Integrated Construction

As stated in the introduction, CIC is viewed as a solution to many of the adverse characteristics of the construction industry, often summed up as a fragmentation of the industry leading to little improvements in its overall productivity (many players with many goals). However implementation of new approaches does not occur without justification, and the barriers for successful integration are significant.

Froese and Russell (1995) discuss design challenges facing the medium sized contractor (who accounts for the largest volume of work in the industry) in consideration of CIC. The discussion calls for an increase in the breadth and depth of existing applications, an ability to work with expertise (captured industry knowledge) and the integration of all applications. Requirements of data representation are needed that allow a hierarchical breakdown based on generic models leading to more comprehensive and richer data sets. All this contributes towards the "critical mass" of supporting tools required before such applications with their promised advantages will be accepted industry wide. Many researchers have proposed hypotheses and strategies for how information technology can successfully contribute towards achieving CIC (Ahmad et al. 1995; Teicholz and Fischer 1994; Breuer and Fischer 1994). Researchers continue to examine the need for changes to industry processes in relation to the adoption of information technologies (Hinks et al. 1997). In general, all approaches call for shared databases of information across the participant boundaries of the industry with an accompanying change in the management style of construction projects. Therefore, to date, it is generally accepted that integrated construction management systems will be conceptually based on a central information source with which integrated applications and the industry participants will interact (O'Brien 1997).

Summary: This research recognizes the ramifications of such strategic changes to the industry upon adoption of CIC. However, the focus is not on the organizational changes required but rather on the supporting tools required for such changes to occur.

2.2. Industry Classification Standards

Industry standards are available for the classification of information and the exchange of standard documents, whether electronic or otherwise. There are existing standard classification schemes whose application is normally on a national level, whose initial development was for the purpose of improving the efficiency of communication throughout the construction process. Examples of these include the MasterFormat (CSI 1995) or UniFormat (CSI 1998) produced by the Construction Specification Institute of the U.S. and administered by the Canadian Standards Association in Canada, and the CI/SfB Construction Indexing Manual administered by RIBA (Royal Institute of British Architects) in the U.K.

For example, MasterFormat, introduced in 1963, was initially based on the nature of construction and the industry at that time. It consists of a single classification table structure of information categories (divisions) that paralleled the disciplines of work (e.g., Division 1000 General Requirements, Division 2000 Site Construction, Division 3000 Concrete, Division 4000 Masonry). Within each division sub-divisions are available to further classify construction information (e.g., 3100 Concrete Forms and Accessories, 33200 Concrete Reinforcement, etc.). However, as the industry changes in aspects such as the structure of its participants, the manner in which projects are delivered, and particularly in the emergence of increased integration of information and electronic communication, these standards are losing there ability to contribute to an efficient communication process.

In light of the recognized importance of standard classification schemes to more sophisticated management systems within the industry, there has been a revamping of the current classification schemes. Evidence of this trend is found through the work of ISO technical committee ISO/TC 59 (ISO TR14177 1994, 1996) and the British derivative of this work, the Unified Classification for the Construction Industry (UniClass 1998). In North America the new trend is towards the product-centered scheme entitled UniFormat. Although UniFormat also serves the industry better in the definition of performance based work (the expected origin of its development), it is also an improved scheme for integrated management system approaches. The Construction Specifications Insitute has also embarked on the Information Integration Initiative in order to support an integrated industry approach (Cassidy 1999).

These new developments within classification efforts recognize the emerging model-based approaches to information integration and are structuring their schemes in multiple tables that address specific types of information groupings. This provides a much more efficient approach to classifying information by allowing the application of multiple tables to provide additional detail for a classification rather than remaining within a single rigid classification table. For example, classification tables exist for "elements" of a project while a separate table is available for "processes" allowing the classification of information from both these perspectives rather than the single tabled approach described above.

Summary: The industry guidelines for classification of information are paramount to structuring and efficiently organizing project management information in a hierarchical representation within an integrated information approach.

2.3. Industry Information Modeling

It is generally accepted that conceptual core models will provide a consistent approach to modeling within a specific application area and increase the capabilities of information sharing and integration among such applications within the architecture, engineering, construction (AEC) domain (Froese 1996). Core models serve as a reference to application specific or domain specific information models and by doing so impute a consistent structure among these specific models that promotes information exchange between applications.

As discussed in Froese (1996), conceptual core information models provide a consistent approach to modeling across related application areas and increase the capabilities of information sharing and integration. This is the approach taken in many efforts within the large-scale information modeling AEC domain. Figure 1.1 demonstrates the layered approach of core modeling. A domain area (e.g., all business within the AEC industry) can be divided into application areas (e.g., structural analysis, project scheduling, etc.).

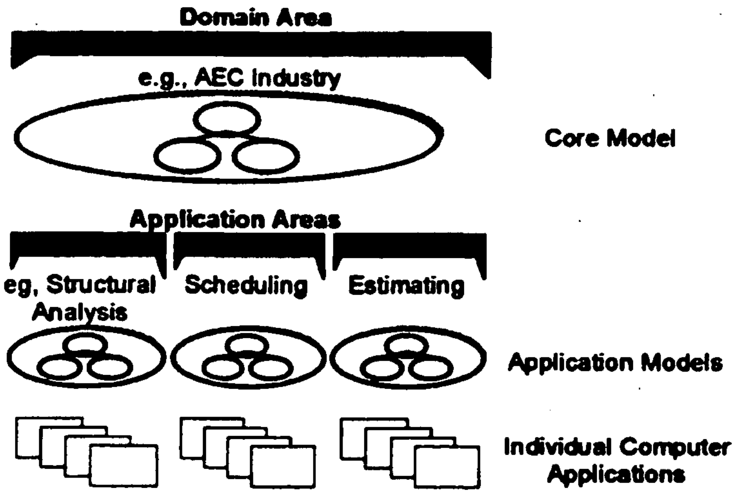


Figure 2.1: Illustration of a layered modeling approach (Froese, et al., 1997a).

The representation of information supporting business activities within a domain follows this same breakdown. The core model at the highest semantic level provides the basis from which more application-specific data models are developed. These application models could be at a level that supports individual computer applications within one or many application areas. The result is a consistent approach to modeling and a mechanism to support information exchange between application areas within the same domain.

Application models represent the information required for a specific application within the core domain. Examples of traditional construction management applications include models for scheduling, estimating, and cost control. New applications such as technology and methods selection, project documentation control, and performance simulation and analysis are also examples of applications for which modeling is applicable. A modeling approach of this nature is emerging to support an overall project concept through the accomplishments of some key research efforts.

The examples presented in this section are efforts in the establishment of information models for the purpose of supporting information integration.

2.3.1 STEP (Standard for the Exchange of Product Model Data)

The ISO standard 10303 or STEP (Standard for the Exchange of Product Model Data) is a standard for the computer-interpretable representation and exchange of product data, with the objective of providing a neutral mechanism to describe the product data through the life cycle of the product independent of any particular system (ISO 1992). Of particular interest is the product data representation of AEC and a subset of AEC, building construction. Due to increasing demands from industry, the STEP effort is moving from a capability of not only data sharing (exchanging data between systems), towards data integration (sharing information among different users and applications) (ISO 1995a). To make this transition several of the fundamental changes being proposed are: (1) to develop core models for specific domains, (2) to implement templates for modeling styles, (3) the establishment of classification schemes, and (4) the development of rules and constraints for mapping purposes. These principles have been applied to the development of the Building Construction Core Model (BCCM) (ISO 1995b).

2.3.2 Industry Alliance for Interoperability

Current efforts within the International Alliance for Interoperability (IAI), a group comprised of industry-based firms, software vendors, and researchers, are working towards the development of Industry Foundation Classes (together forming a core reference model) for exchanging data between computer systems within the AEC industry (Autodesk 1995) (IAI 1997). There is in fact quite a synergy between individuals involved with both the STEP and IAI initiatives. Specifically applicable to this research is the work being produced by the domain committee focusing on project management related aspects of the Industry Foundation Classes (Froese and Yu 1998, 1999).

2.3.3 Project Models

Various research efforts have contributed to the research noted above, such as the GRM's (General Reference Model) a predecessor of STEP (Reschke and Teijgeler 1994). Recognized advancements to project model development have also been cited to the modeling work of Froese (1992) and Bjorg (1992).

The COMMITT (Construction Modeling Methodologies for Intelligent Information Integration) project (Rezgui, et al. 1996) is based on the work of the previous ICON (Information Integration for Construction) project (Ford et al. 1994). The project produces individual models to describe different perspectives of project information (e.g., a construction planning object model) based on sound information technology frameworks, a slight alternative to a meta-model approach.

Other research project's have also recognized a layered modeling concept moving from generic meta-models, to conceptual models, to industry reference models, and finally to specific project models. An example is the work performed at VTT Building Technology focusing on the development of models and tools for the improvement and re-engineering of construction processes (Hannus and Pietilainen 1995, and Hannus 1996).

2.3.4 Process Models

Existing models (perhaps better characterized as descriptions) of industry processes, such as the Project Management Institute's (PMI) Body of Knowledge (BOK) document are also available for reference (PMI 1996). Additional examples can also be found in the works of Sanvido (1990) and Gillespie (1998).

2.3.5 Other Models

Efforts directed toward other AEC disciplines such as COMBINE (Computer Models for the Building Industry in Europe) (Augenbroe 1995), which emphasizes design practices, are also useful sources of reference. As are the models developed as a result of integrated systems development, noted below.

Summary: This research is interested in the representation of the information required for a comprehensive project description (i.e., products, processes, resources, etc.), as well as, a focus on the integration capabilities offered by the application of standard models. Essentially STEP's BCCM "as is" model was the starting point of model development with consideration given to on-going modeling efforts and approaches within IAI.

2.4. Automated Planning

Automated planning is not a new topic. Past approaches have focused on artificial intelligence (AI) solutions (Levitt et al. 1988) and were initially centered on basic information objects and low level reasoning to derive project components and to generate a time scaled activity sequence. Although the approaches have been criticized for their inability to effectively handle the size and complexity of actual projects, they have most definitely provided a thorough foundation describing the structure of construction planning. The references that follow are prominent efforts in this area of research.

2.4.1 CONSTRUCTION PLANEX

CONSTRUCTION PLANEX (Zozaya-Gorostiza et al. 1989; and Hendrickson et al. 1987) is an expert system addressing all aspects of construction planning from identifying activities required for given design elements to selecting technologies and construction methods, and estimating durations and costs. The system requires detailed design information describing the structural elements of a building. Knowledge sources are used for selection of construction methods and assignment of durations and costs. The knowledge is essentially activity based and cost models are described as simple in nature. Precedence relationships are pre-defined and the aggregation capabilities for activities are limited. The focus of the work is primarily on the precedence and scheduling aspects of construction planning.

2.4.2 Sequencing Knowledge

Investigation into the formalizing of sequencing knowledge for construction scheduling is described by Echeverry et al. (1990 and Echeverry 1991). It is proposed that sequencing knowledge can be captured, and divided into one of four groups: physical relationships, trade interactions, path interference, and code regulations. A further breakdown of each group is performed and the resulting application of the knowledge is to serve as an aid for scheduling construction projects.

2.4.3 Other Research

GHOST (Navinchandra 1988), OARPLAN (Darwiche et al. 1988; Winstanley et al. 1993), SIPEC (Kartam and Levitt 1990, Kartam et al. 1991), ACP (Waugh 1990), and Builder (Cherneff et al. 1991) are all further research examples of AI approaches to the automation of construction planning. Included in the next section are some current extensions to these works.

Summary: For this research, the importance of automated scheduling research performed to date is the formalization of construction planning knowledge and the approaches taken to capture this knowledge for future use. Equally important are the methods used for grouping and classifying this knowledge and its interrelationships.

2.5. Assisted Planning and Integrated Systems

The following efforts are in line with: a systems approach to planning with a movement towards integration of applications; a higher or richer level of information representation; and the use of templates or libraries of knowledge structures which can be aggregated, easily accessed and applied. The shift is towards information technologies as opposed to a focus on AI solutions. These research projects are more closely related to the end result of computer-assisted construction planning and do incorporate some but not all of the approaches under consideration.

2.5.1 Integrated Construction Planning System (ICPS)

The integrated construction planning system (ICPS) proposed by Yamazaki (1993, 1995) focuses on formalizing the representation of the planning process as an approach to CIC and promotion of interactive cooperation between distinct stages of planning. The research defines building system planning and construction system planning together as an interface between design development and construction planning. Building system planning, a function of the design process, specifies the functional requirements of building spaces, develops subsystems and components, and defines functional requirements of the subsystems and components. The construction system planning evaluates the results of building system planning and proposes construction processes and methods for the subsystems and components based on requirements and constraints as the plan progresses through formalized stages of planning (concept, fundamental, and practical).

The research recognizes the differing approaches of a top-down approach, based initially on a product model and deriving processes, as well as a bottom-up approach focusing on the process model (construction activities). A project modeling system based on product and process models allowing middle-up-and-down approaches (comprehensive planning) is proposed based on an object oriented paradigm.

The ICPS also uses a hierarchical representation of each distinct model identified, however no indications are given as to how the mapping is defined between different levels in each model. The research also defines distinct planning stages and recognizes the need to support an evolution of the construction plan. The storing of information in a general planning model suggests that some sort of information type library is considered.

2.5.2 Automated Building Realization System (HISCHED)

Various components of an integrated automated building realization system have been investigated at the Israel National Building Research Institute and are based on a project model (Warszawski and Sacks 1995, Warszawski and Shaked 1994). One of the components of the system is HISCHED (Shaked and Warszawski 1995), described as a knowledge-based expert system for the construction planning of buildings. The approach of this research is that project models should be tailored to specific building types. The research is described in terms of design and construction of orthogonal, multi-story buildings with uniform floor plans. The project model is composed of a product, resource, and activity models. A structure of building process stages are defined with specified input requirements and expected output results with the building project model data base growing in detail at each stage.

The approach begins with the identification of building spaces and their required functional systems. Technological solution objects are then provided for each functional system object, and prompt definition of the required work assembly objects and element objects. Basic activity objects are then defined to install assemblies and elements. A hierarchical breakdown requirement is recognized for assembly, element, and activity objects. As well, mapping issues between different hierarchical levels of objects and duplication of fulfillment of functional requirements by single elements is addressed. The computational solution is described as a "parametric assembly" or "adoption of templates" solution. The premise is that the templates are stored in libraries outside the calling program with complete definitions able to serve at any stage due to their conformance to the project schema and defined hierarchy.

The approach requires a project model structured in such a way that it can facilitate the addition of detail throughout the project life cycle, however the manner in which the hierarchical mapping is performed is not very clear. A restriction of parallel major classes of space, product and activity classes is alluded to which may address the mapping issue at one level, however other fundamental objects such as cost and resources are not discussed. The approach is quite rigid in its structure and limited in its application.

2.5.3 CADCIMS

A more practical focus on the management of product and process information is taken by researchers at the US Army Corps of Engineers Construction Engineering Research Laboratories (USACERL). The system developed, CADCIMS (computer aided design and construction management information system) (Stumpf et al. 1996), links several project management software applications through a relational database. Building components and process information are linked by allowing a grouping of components to be linked to selected activities, essentially an activity view. The systems goals are listed as allowing integration, multiple views and ability to handle changes in information. This initiative calls for future research into a structured and efficient information storage to help improve the information transfer.

2.5.4 Construction Method Driven Scheduler

CMD Scheduler is a prototype system developed with the intention of addressing the requirements of applying knowledge-based scheduling and planning system as industry applications (Aalami and Fischer 1996, Fischer and Aalami 1996). Central to this approach is the use of construction method information, which is captured in a method model and re-used in the development of estimates and schedules.

In the context of an integrated construction management system, the researchers note that there are no current mechanisms that "automatically" and "dynamically" link design information to construction planning and scheduling information. With the assumption that the user has the knowledge to make decisions regarding the construction methods applied, the prototype system addresses this link in support of the rapid development of estimates and schedules.

A 3-dimensional CAD product model provides input to the model, while process knowledge is available in the form of construction method templates. The construction method templates contain information regarding their application domain, the construction activities the method is composed of, the sequencing of these activities, the resource requirements, and the objects that each method is applied to. A plan is developed by starting from a high level process-product pairing and applying the applicable methods that introduce a refined level of process-product information. These steps are repeated to the level of detailed required.

2.5.5 IT Support for Construction Planning

Jägbeck (1994, 1998) describes the development of several prototype applications in support of construction planning through the incorporation of a project's product information and lays out the conceptual framework for an integrated application that addresses construction planning.

Again, the overall framework recognizes the need to integrate design information and construction knowledge in a planning application (product-process integration). The prototype application MDA Planner defines "methods" as stores of generic construction planning knowledge. A "plan" is developed by adopting a method in consideration of existing preconditions stored in the method template along with the applicable products and required resources. A second prototype, PreFacto, focuses on the conversion of "plans" to "methods" and the reapplication of "methods" to products to form new "plans".

The eventual extension to the planning systems described is a system that supports a link between design and planning through the use of an integrated information model and the case based reasoning of construction planning knowledge stored in template structures.

2.5.6 REPCON Automated Scheduling

The evolution of the research construction planning system REPCON (Russell and Wong 1993) continues with the conceptual description of a module in support of automated scheduling. The module is intended to support the creation of draft plans and schedules through the application of expert systems and large building blocks of predefined scheduling knowledge (Chevallier and Russell 1998). This research stresses the development of an automated scheduling tool that is functional in today's construction management environment.

The approach described also recognizes the importance of an integrated product and process view of project information (Russell and Chevallier 1998) and a need to re-use past construction planning knowledge. The approach expects the user to guide the development of a plan with the option of applying the support of a simple set of rules in dealing with the scale of a project. The rules are applied to attributes such as scale, site context, and contractual requirements in the planning process. Planning knowledge is stored in the form of "activity fragnets" that consist of the activities required to construct part of the defined product complete with their precedence relationships and logic.

Summary: An approach supporting both aggregation and specialization of planning information is required. This must be based on an object-oriented paradigm (allowing a rich representation of project information), with a standard framework (model) applicable to a wide range of construction projects. Planning occurs across multiple hierarchical levels, and therefore the approach must allow a progression of construction plan development. The concept of storing and retrieval of planning knowledge through the use of templates is the approach adopted.

2.6. Case Based Reasoning

The original hype of "case-based reasoning" began with Riesbeck and Schank's (1989) Inside Case-Based Reasoning, a text built on the flourish of Case-Based Reasoning research performed at Yale University in the late 80's. Today case-based reasoning is presented as a methodology for dealing with large quantities of information (Watson 1998) where various technologies (nearest neighbour, induction, fuzzy logic, SQL, etc.) are applied to the original algorithm.

As applied to planning, people do not construct plans from first principles, rather they try to find the best previous plan and adapt it to the current situation. A "case-based reasoner" solves new problems by adapting solutions that were used to solve previous problems with its elements of a case library and methods of storage, indexing, matching, and adaptation.

The basic algorithm that is applied is: 1. input the problem, 2. find the cases similar to the current problem (characterize the input problem, retrieve cases with matching features, pick the best match(es)), 3. adapt a previous solution to fit the current problem. In addition, concepts are required to support the ever-changing knowledge base that the case-based reasoning uses. To support these dynamic memory requirements, common information management concepts of inheritance, aggregation, specialization, and attribute definition are required.

Applications of case-based reasoning to engineering and construction include design (proportionally much higher), contracting, the application of building regulations, scheduling, and estimating. The following research is noted due to its similarity with the approach adopted for CACP, while bringing to light the issues involved with this application of case-based reasoning.

2.6.1 Design and design selection

Various systems that apply case-based reasoning to support design assistance and design automation are discussed in Maher and Gomez de Silva Garza (1996, 1997) along with the difficulties of their implementation. Voß (1994) describes a retrieval method for the selection of appropriate designs for building projects in the context of reusable parts rather than entire designs. Sycara et al. (1992) apply CBR to act as a designer's assistant in the design of mechanical devices. While Bilgic and Fox (1996) discuss case-based retrieval as used in engineering design within the context of an enterprise system. Finally, Flemming et al. (1994) present case-based reasoning in support of early phases of building design. Although the area is not mature enough to lead to a generally accepted approach, there are many recurring themes noted that are equally applicable to the application of case-based reasoning to construction management.

In general, the two goals of re-using previously stored design information are to provide access to a large memory of past solutions, and act as an aid by providing designers with an initial solution available for editing by the system or user. Addressing these goals for construction planning is also the motivation for CACP. The first step is a solution to the representation of complex cases, where generalized knowledge and experience must be formalized. The common approach is to represent this information in an object-attribute schema while also maintaining categories for the various attributes (e.g., relation, function, behavior, and structure). Also, the reuse of solutions is applied across many phases of a projects design cycle (e.g., conceptual design to detailed design) and therefore retrieval must be available across a problem that is decomposed in a hierarchical structure.

Potential solutions must have a dynamic indexing mechanism because the retrieval process is iterative (i.e., the level of representation of the problem varies). Throughout the retrieval process the user is quite involved with the pruning process as a design develops and additional information is added. Therefore, using a hierarchical approach (relation attributes) allows access to potential sub-solutions.

A retrieval process is normally simplified by establishing the context of the problem through definition of a set of constraints. The context issue is dealt with up front by making it an explicit part of the query. Then, for each retrieval, a clustering of cases is performed based on the context. The method of retrieval and the means of indexing during retrieval are central to the process. First, the context of the information to be retrieved is established, then, the attribute-value pairs that will be used for retrieval are selected.

2.6.2 Other CBR applications

Case-based reasoning has also been considered for application to other areas within the AEC industry. Yang and Robertson (1995) provide a framework for interpretation of building regulations, through abstraction hierarchies of legal rules from statutory regulations and previous relaxed cases. This example fits into the domain of case-based reasoning as applied to legal applications. Ng et al. (1998) apply a case-based reasoning tool and address case representation, indexing and retrieval, and adaptation for contractor pre-qualification by guiding the user through the processes of criteria formulation, screening and reviewing, and final assessment based on financial and performance issues. Both examples show the potential of case-based reasoning as an effective aid to applications within the AEC industry.

2.6.3 Construction Planning

Perhaps closest to the domain of CACP is the research performed by Dzeng and Tommelein (1995, 1997) towards the development of CasePlan and Tah et al. (1998) with their development of CBRidge, both prototype case-based reasoning system for construction planning. Although these prototypes are not based on an integrated view of project information, they have considered the basic elements covering a projects products, processes, and resources. Both research efforts support the concept of applying case-based reasoning to construction planning.

Summary: The knowledge component of the CACP prototype is based on a case-based reasoning approach. As with other applications of case-based reasoning, the greatest challenges are in the proper representation of information that forms a case to ensure successful retrieval and adaptation. The application of case-based reasoning for CACP is applied to demonstrate proof of concept for managing stored information.

2.7. *Hierarchical Interfaces*

Construction management information is inherently hierarchical in nature, implying information in a tree structure. This can be attributed to the unique characteristics of the industry mentioned previously (many players with many goals) which in terms of the information requirements translates into many views at a variety of levels of detail. The hierarchical structuring of information is one of the reasons behind the need for effective classification systems discussed previously and it is also a primary concern when developing the interface for any construction management application.

One of the most important considerations for a software interface that deals with large hierarchical information structures is the "focus+context" issue (Lamping et al. 1995). The "focus+context" issue is the term coined to describe the trade-off between showing sufficient detail in a hierarchy (focus) while maintaining a sense of where one is in this same hierarchy (context). 2-dimensional techniques that have been proposed include fish-eye views that allow the user to zoom in on portions of a hierarchical display while maintaining a perspective on where they are located in the hierarchy (Sakar and Brown 1992). The problem with 2-dimensional approaches is that the growth in the number of nodes for a single tree is exponential making the layout of any tree very difficult. To reduce the complexity of a single tree, researchers have looked for ways to conveniently partition a single tree into multiple trees based on the properties of nodes (leaves on a tree) (Koike and Yoshihara 1993).

The advances in this research area of hierarchical representation and manipulation are being made with 3-dimensional solutions that are employing techniques such as cone structures or box structures to represent the third dimension of a hierarchy. However, solutions that require only modest computational needs are only now emerging (NVision 1999) and therefore it is more appropriate to look at current solutions and examples to 2-dimensional tree representation.

A combination of a tree view and detailed information window is the most common example of an effective 2-dimensional hierarchical interface which allows the user to expand or collapse branches of the tree structure while clicking on a tree node to view detailed information relevant to each individual node. Microsoft's Windows Explorer is an example of this approach with its folder and file tree structure and detailed window of file information.

Summary: The application's interface will follow the trend of current 2-dimensional representations of hierarchical information while considering methods of reducing the complexity such as partitioning and the combined use of a tree view and detailed view.

2.8. Implementation and Practical Project Management Tools

An investigation into the current state of industry-generic project management software (e.g., Microsoft's Project, Primavera Systems' Sure-Trak, Scitor's Project Scheduler, etc.) and its level of support in the storage and reuse of past planning knowledge was completed. This was a result of several consulting projects undertaken by the Construction Technology Centre Atlantic Inc. (CTCA), and lead by the author. The intent of the investigation was not to compile an exhaustive list of the products available in support of project management functions but rather to gain an overview of existing capabilities and contribute to defining the functionality required of CACP. Concepts related to the capture and reuse of planning knowledge continue to be examined under ongoing consulting projects.

2.8.1 Generic Planning Software Applications

In summary, the general trend among the project management applications examined, was a concerted effort towards: increased capabilities in support of a multi-project planning and control, integration with complementing communication applications, and a shift towards a client/server type environment. The reuse of past planning information, if supported at all, is generally limited to a simple storage and recall of scheduling templates. The most sophisticated of the applications examined, makes use of a "wizard" that leads the user through a brainstorming session to initially identify project phases, goals, resource, assignments, and tasks—which are selected from an organized collection of template knowledge. At the end of a "wizard" session the user has the beginnings of a plan. However, the primary perspective for these applications is scheduling (time planning) only.

2.8.2 Project Management for Information Systems

In addition to the generic project management applications examined, several plan development applications for the Information Systems (IS) domain were also examined (i.e., System House's Transform, LBMS's Process Engineer, and Xpert Corporation's PlanXpert). A common trait among these planning applications is the reuse of planning knowledge that is stored based on industry methodologies. Using an analogy to the construction industry, these methodologies could be thought of as different contracting approaches (e.g., fast tracked design build construction). From a scheduling perspective (a hierarchy of tasks), these IS tools offer additional information such as task deliverables, resource requirements, and precedence relationships. More detailed information—which might be considered a step towards the capture of planning knowledge—includes guidelines for estimating task duration based on suggested resources and definable project parameters (e.g., size and complexity of the project). These parameterized estimates and the results also serve as benchmarks for future projects upon completion.

For the IS tools, a plan is also supported by information (in a variety of multi-media formats) attached to appropriate tasks, that provides an explanation of how the task is to be completed. For example, if the task is to define the data requirements of an information system, then this task may be supported by an explanation on how to create an entity-relationship diagram and perhaps even contain a link to the software tool supporting its creation. It is these types of advanced support tools' concepts that are proposed for total project systems

Summary: The trends in current software supporting the planning process are to increase integration and communication with other project management tools while directly supporting the planning process through the provision of guidance (some in the form of templates) throughout the planning process. These trends also match with the objectives of CACP.

2.9. Summary

The following statements describe the starting point for this thesis and point the way for the research thrusts undertaken.

This research recognizes the ramifications of such strategic changes to the industry upon adoption of CIC. However, the focus is not on the organizational changes required but rather on the supporting tools required for such changes to occur.

The industry guidelines for classification of information are paramount to structuring and efficiently organizing project management information in a hierarchical representation within an integrated information approach.

This research is interested in the representation of the information required for a comprehensive project description (i.e., products, processes, resources, etc.), as well as, a focus on the integration capabilities offered by the application of standard models. Essentially STEP's BCCM "as is" model is the starting point of model development with consideration given to on-going modeling efforts and approaches within IAI.

For this research, the importance of automated scheduling research performed to date is the formalization of construction planning knowledge and the approaches taken to capture this knowledge for future use. Equally important are the methods used for grouping and classifying this knowledge and its interrelationships.

An approach supporting both aggregation and specialization of planning information is required. This must be based on an object-oriented paradigm (allowing a rich representation of project information), with a standard framework (model) applicable to a wide range of construction projects. Planning occurs across multiple hierarchical levels, and therefore the approach must allow a progression of construction plan development. The concept of storing and retrieval of planning knowledge through the use of templates is the approach adopted.

The knowledge component of the CACP prototype is based on a case-based reasoning approach. As with other applications of case-based reasoning, the greatest challenges are in the proper representation of information that forms a case to ensure successful retrieval and adaptation. The application of case-based reasoning for CACP is applied to demonstrate proof of concept for managing stored information.

The application's interface will follow the trend of current 2-dimensional representations of hierarchical information while considering methods of reducing the complexity such as partitioning and the combined use of a tree view and detailed view.

The trends in current software supporting the planning process are to increase integration and communication with other project management tools while directly supporting the planning process through the provision of guidance (some in the form of templates) throughout the planning process. These trends also match with the objectives of CACP.

Chapter 3: Project Management Data Model

All effective information technology management systems must be based on a well-defined data model. The importance of such a data model to this research has been mentioned several times in the previous chapter. This chapter explains in detail the relevance of the data model, describes the steps taken during its development, and defines the model in its entirety. Preliminary steps in this data model development, such as identifying existing models for comparison, were performed in collaboration with other TOPS researchers at the University of British Columbia. The author then completed each data model development step in the context of this research.

3.1. Purpose of the Model

The foundation of the TOPS approach is basing systems on common data models of construction project information. As stated in Froese et al. (1997), the significance of the common data model is that it provides the primary mechanism: for structuring data within TOPS applications; for exchanging information among TOPS components; and for interacting with other computer applications through international data standards.

3.1.1 Scope of the Model

The intended scope of the data model covers the discipline of construction management, and perhaps the broader scope of project management, as applied to AEC projects. However, the depth required to support specific applications such as cost estimating or resource management is not the intent.

7:5
40.
11:20.

The reason for this breadth of scope stems from a need to support not only areas of project management that are currently well-supported by computer tools but also applications within TOPS that are intended to address new areas of computer support within project management. Currently, documents relating to virtually all of project management can be tracked within computer systems, requiring at least some information about the documents (if not all of their contents) to be structured in the data model.

7:50
7:00

The approach is to develop all project management application areas within one overall project management application model for the present. This is an alternative to separating the information model into distinct application models. A larger model promises easier integration across applications areas within project management, but leads to greater difficulties in developing and maintaining the model. Partitions to the overall model are still possible in future revisions, while still maintaining the layered modeling approach. A certain amount of depth is added for the specific application of CACP.

3.1.2 Relationship with International Data Standards

The PMDM adopted emerging international data model standards that were available during this development, such as STEP's Building Construction Core Model (BCCM) and the IAI Industry Foundation Classes where they already address the appropriate bodies of information. These models cover much of the breadth of project management functions, but as expected of core models, they fall short of developing the necessary depth within specific application areas. The Industry Foundation Classes have a section that covers project management, and several application areas are under development such as scheduling and estimating (Froese and Yu 1999). A major objective of the project management data model for TOPS, beyond supporting the TOPS system itself, is to contribute to these development efforts, an ongoing process.

3.1.3 Modeling Methodology

The development methodology used for the PMDM draws from several modeling methodologies. The main technique is to carry out some form of functional analysis that identifies the functional requirements for the data model, then to devise the data topics required to meet these functional requirements, and finally to fully detail the object definitions that describe each data topic.

This approach was initially applied with the scope of TOPS in mind and later refined specifically for CACP. The functional analysis consists of developing a hierarchical listing of functional capabilities required of the systems that the data model would support. Because the goal is to develop a common data model that serves as a unifying core of many individual applications, this is a more abstract process than if the functional analysis were being performed for a single specific computer application. The initial analysis consisted of defining the functional requirements in the form of functional categories (topics) that the system and thus the data model should be capable of working with (a top-down approach) (Table 3.1). A subsequent analysis defined the functional requirements from the perspective of the specific documents that are commonly used in the construction industry (a bottom-up approach). The result was categories of data types required to support the functional requirements. These data topics (categories) correspond to the units of functionality in the STEP methodology. The collection of all of the data topics makes up the scope of the data model. The resulting data model was then used for development of the PMDM for CACP.

3.2. Model Development

As mentioned, CACP takes a broader view of planning than what has been typically tackled by construction plan generating tools. The scope of planning supported by CACP is defined in terms of the Project Management Institute's Guide to the Project Management Body of Knowledge (PMBOK) (PMI 1996) and its definition of knowledge areas for describing project management processes. As a further refinement to the scope of CACP, the project phases to which the planning will apply begin with the conceptual phase and end before the development of a detailed construction plan. Again, CACP is used to initially populate a construction plan with information based on past experience. Its purpose is not to replace existing detailed project management applications, but rather to provide a stepping stone for subsequent detailed planning and controlling applications.

3.2.1 Processes Supported

The supported planning processes are defined using the PMBOK's terminology. CACP is intended to support planning from the perspectives of: scope, time, cost, performance (quality), and organization. Therefore, support is extended past the representation of core planning processes of scope, time, cost by adding knowledge that is usually implicitly included in the planning processes (i.e., performance and organization). The results of this analysis of the planning processes supported by CACP are demonstrated in Figure 3.1.

A process modeling structure similar to the format applied in the $IDEF_0$ (NIST 1993a) methodology is used to describe the planning processes. As is the case in $IDEF_0$, the outputs, inputs and constraints are identified as flows leading to and from a given process. Mechanisms, however, are usually synonymous with the resource requirements of the process, and in the case of construction management processes, identify the personnel involved. It was found more useful to identify the potential techniques that the processes may use. Also, in the case of construction management sub-processes, there is not necessarily a sequential execution and, therefore, it is not as useful to display them in the common $IDEF_0$ node format (i.e., expanding a process into sub-process and showing their internal flows). An additional modification is the concern with the source of flows—specifically, whether they originate from, or contribute to, a general industry information source or a specific project information source. In the case of CACP, the general information corresponds to past captured planning knowledge while project information corresponds to the current project being planned.

Figure 3.1 shows the highest level process of *construction planning*. Using the modified $IDEF_0$ notation the figure shows that, at an overview level, the *inputs* originate from the *general information* (e.g., historical information, work breakdown structure, etc.) and the planning process produces *outputs* contributing to the *project information* (e.g., activity list, resource requirements, etc.). *Constraints* initially originate from the *general information*, but they also draw on the *project information* as a plan is defined when knowledge about the planning environment is added and the constraints become more focused. Figure 3.1 also shows the sub-processes within the scope of CACP. The outputs identify: the potential internal flows within construction planning; how these internal flows add to the project information; how they are used in other sub-processes; and how they add to the constraint knowledge.

Not all mechanisms and techniques shown are necessarily supported by CACP (e.g., decomposition and template use are, while mathematical analysis and simulation are not). However, the data requirements for all techniques must be supported at least at an abstract level of detail. The resulting functional categories of this planning process analysis are displayed in Table 3.1. These functional categories served as the starting point for the elaboration of each information object towards development of the PMDM required for CACP. The modeled objects went through several iterations and were defined and organized into units of functionality as presented in the next section.

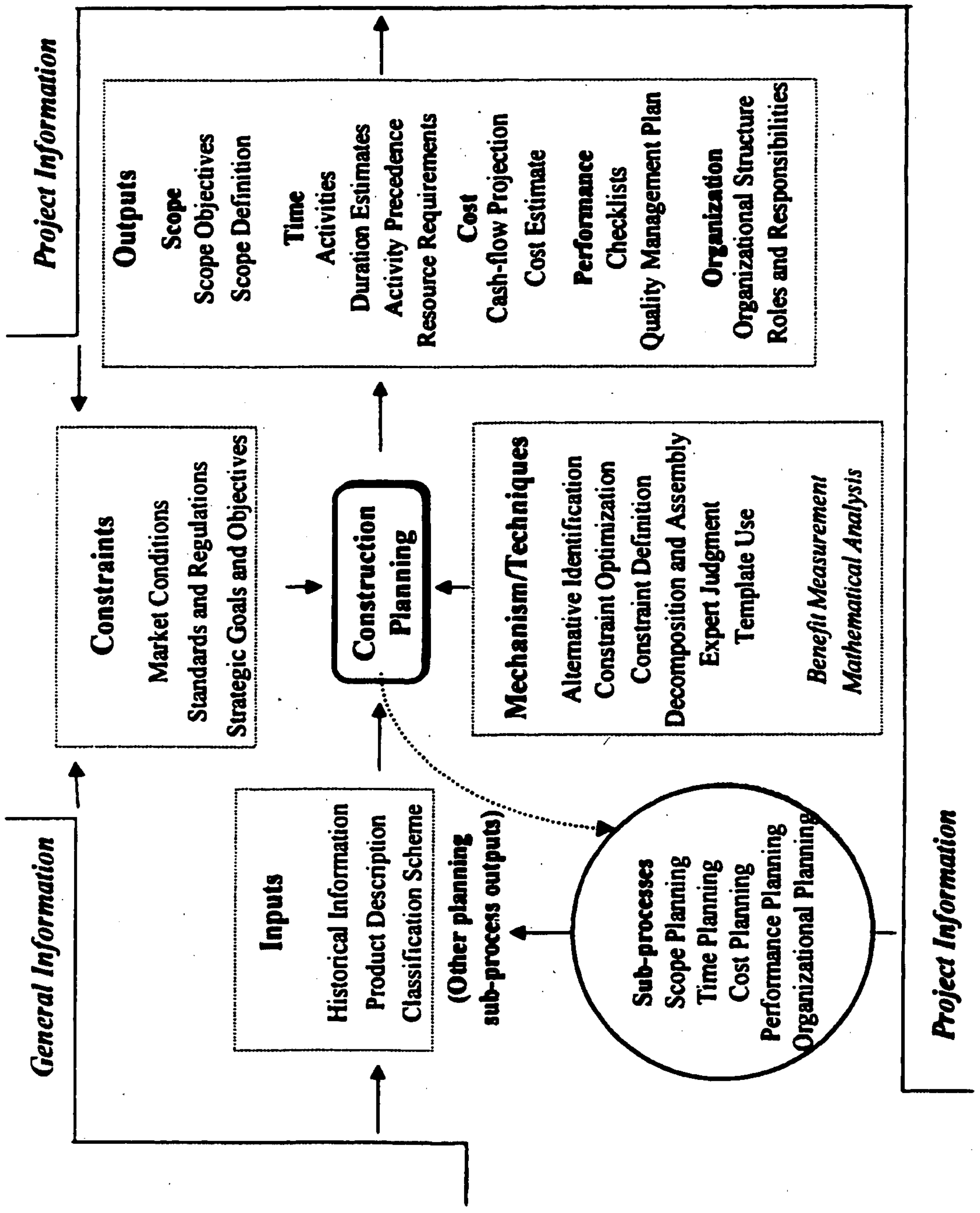


Figure 3.1: Planning processes supported by CACP.

Table 3.1: Functionality categories for CACP.

General	Time	Performance (Documents)
Classifications	Activity	Codes of Practice
Lists/Indexes	Activity Sequence	Conformance
Meta-Model	Constraints	Regulations
Object History	Process	Specifications
Root and Kernel	Schedule (Document)	Standards
Scope	Timing and Dates	**Organization**
Drawing (Document)	**Time (Resource)**	Organizational Structure
Product	Resource	People & Organizations
Product Design	Resource Allocation	Participant Roles
Product Representation	Resource Availability	Responsibility
Cost	Resource Cost	**Documentation**
Cost	Resource Productivity	Contracts
Cost Information	Resource Requirement	Control
Cost Type	Resource Type	Drawings
Estimate (Document)	**Time (Methods)**	Estimate
Estimate Item	Construction Conditions	Purchases
	Method	Requirements
	Methods Plan	Schedule

3.3. *Adoption of Existing Data Models*

The development of the data model began with the adoption of the Building Construction Core Model (BCCM) version Str411 (ISO 1995b) and followed several versions of this effort including versions T100 and T200 (ISO 1996). As the focus of those involved in the development of the BCCM shifted to efforts within International Alliance for Interoperability, so did the tracking of changes to the core modeling efforts. The final models examined for adoption of modeling constructs was International Alliance for Interoperability's Industry Foundation Classes (IFCs) versions 1.5 and 2.0 (IAI 1997, 1998).

As its name suggests, the BCCM is intended as a core model for building construction and therefore has a narrower scope than that of the Industry Foundation Classes, which are intended for the entire AEC industry. The Industry Foundation Classes also contain several layers which include not only a layer for defining the core model but also layers that support the "core layer" (the "resource layer defines basic concepts such as data types and units of measure) and expand the "core layer" in areas to support specific applications (the "interoperability layer" and "application layer"). Therefore, detail was added to the PMDM initially based on the BCCM, while not all Industry Foundation Class constructs were determined to be relevant or the complexity provided not required, due to the specific applications they support. Where simplifications are implemented, it is for the most part the adoption of a default list provided in place of a more complicated object structure. For example, a default list of "product systems" is used rather than defining these systems as objects within the model. Distinctions between the two base models (Building Construction Core Model and Industry Foundation Classes) and the resulting PMDM are provided in the following section.

The PMDM model also benefited from a precursor project that consisted of the development of a data model to support a commercial Facility Management Turnover System (Froese, et al. 1997a) in which the author was a principal in developing. This model was also based on the latest developments within IAI.

3.4. *Modeling Issues, Solutions, and Approaches*

This section describes the issues that have been addressed throughout the development of the current version of the PMDM. Each issue corresponds to a "Unit of Functionality" (UofF), the term adopted to define the partitioning of the data model for the purpose of more easily describing and defining it. The information structure adopted within each UofF is shown in the figures that follow each section along with an explanation of the intended solution it offers and how this solution compares with other modeling efforts.

Although the model is representative of an object model, the model was developed with a customized CASE tool built in Microsoft (MS) Access and later implemented in MS Access. Therefore, the diagrams are displayed using IDEF1X (NIST 1993b), a semantic data modeling technique that has a simple representation and is convenient to produce from an MS Access environment. Produced with Visio Professional~~, the diagrams~~ include ~~standard~~ attribute notation to identify the relationships between entities (Figure 3.2). The notation (FK) is used to indicate a "foreign key" (identifies the primary key attribute of another entity contributed by a relationship), while (IE) denotes "inversion entry" (a non-unique access identifier for an entity).

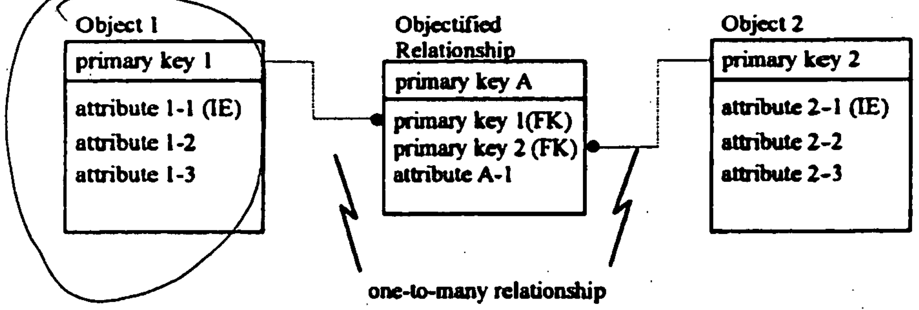


Figure 3.2: Legend for IDEF1X diagrams.

Referring to Figure 3.3, each of the information objects is represented as a table (box) with the primary key listed first (e.g., project object id). The lines that connect each box represent the one-to-many relationships between tables with the nodes representing the many. Foreign keys denote a link to the primary keys through the unique identifier of a related table (e.g., Products: project object id) while the inversion entries denote an alternative identifier within a table (e.g., Products: identification). The common modeling techniques adopted include the use of simple data types (e.g., text, numeric, selection list, etc.), objectified relationships (relationships between objects can have attributes), and abstract objects (not all objects can be instantiated). The table in Figure

3.3 for *kernel object types* is an example of a selection list that takes the form of a single field table. Figure 3.3 contains the first examples of objectified relationships with the tables representing *Product Inputs*, *Product Outputs*, and *Resource Usage*. Using this structure makes it possible to include additional attributes for relationships (e.g., Resource Usage: quantity).

The description of the data model that follows does not provide a detailed explanation of the PMDM's meta-model. The integrated graphical representation of the PMDM is provided in Appendix A. The integrated data model in Appendix A also distinguishes between information classes that were adopted from existing core models (23 classes) and those that were added to support CACP's information requirements (32 classes).

3.4.1 Kernel objects

In keeping with the layered core model described earlier, there are fundamental objects and relationships upon which all project information is built and there is a requirement of a parent object (root) to which all other modeling features (attributes) are attached.

As per the Building Construction Core Model (BCCM), the root object is the abstract *Project Object* and the kernel objects consists of the *Process*, *Product*, *Resource* and *Control* objects (Figure 3.3). All project information is described by elaborating on one of these kernel objects. The *Projects* object distinguishes each kernel object based on a specific project. The *kernel object types* refer to kernel objects that form part of a library of general project information. Each kernel object may rely on a specific *kernel object type* for the definition of an inherited set of attributes.

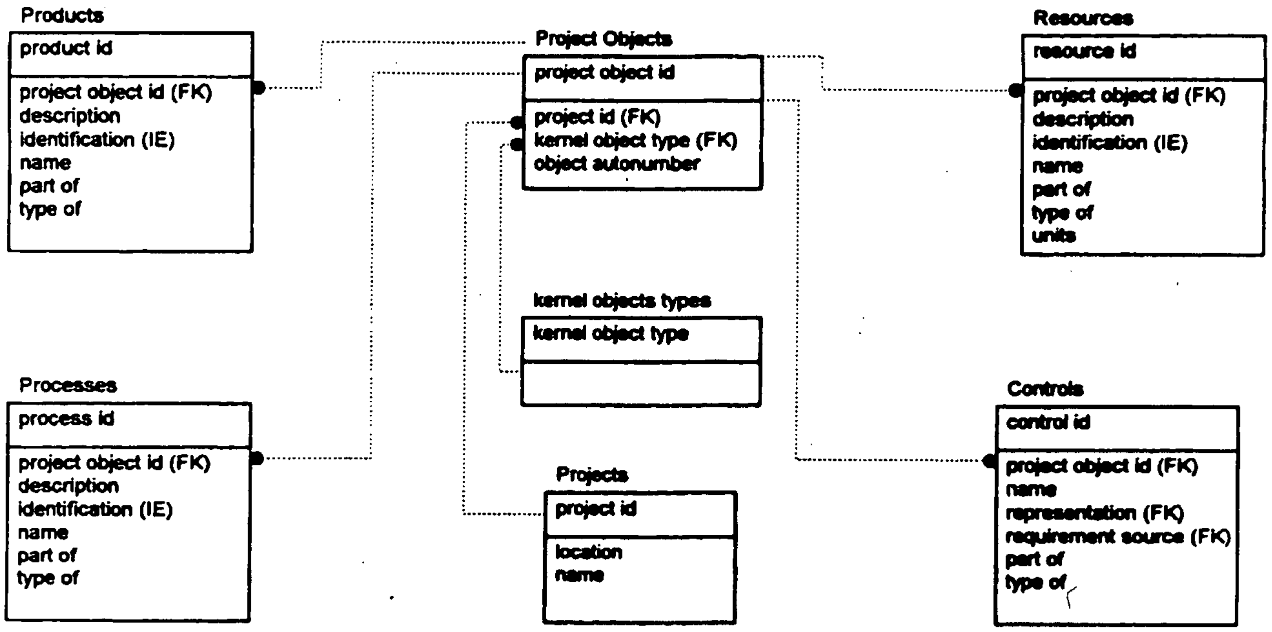


Figure 3.3: Kernel objects IDEF1X entity relationship diagram.

The Industry Foundation Classes defines several additional objects at this level of the model: IfcModelingAid, IfcProxy, and IfcDocument. The first two objects are beyond the scope of the PMDM, while the concept of a "document" and its representation in the PMDM is described under the *Records* UofF. The Industry Foundation Classes also define the concepts of "occurrence" and "extended" properties at this level of the model, these concepts are dealt with under the *Attribute Sets* UofF.

As depicted in Figure 3.4, the objectified relationships between the kernel objects *Products*, *Processes*, *Resources* are similar to the constructs used in the Industry Foundation Classes. *Products* can act as input or outputs to a *Process* while *Resources* are used in *Processes*. Again, *Controls* are treated slightly different as described below under the *Controls* UofF.

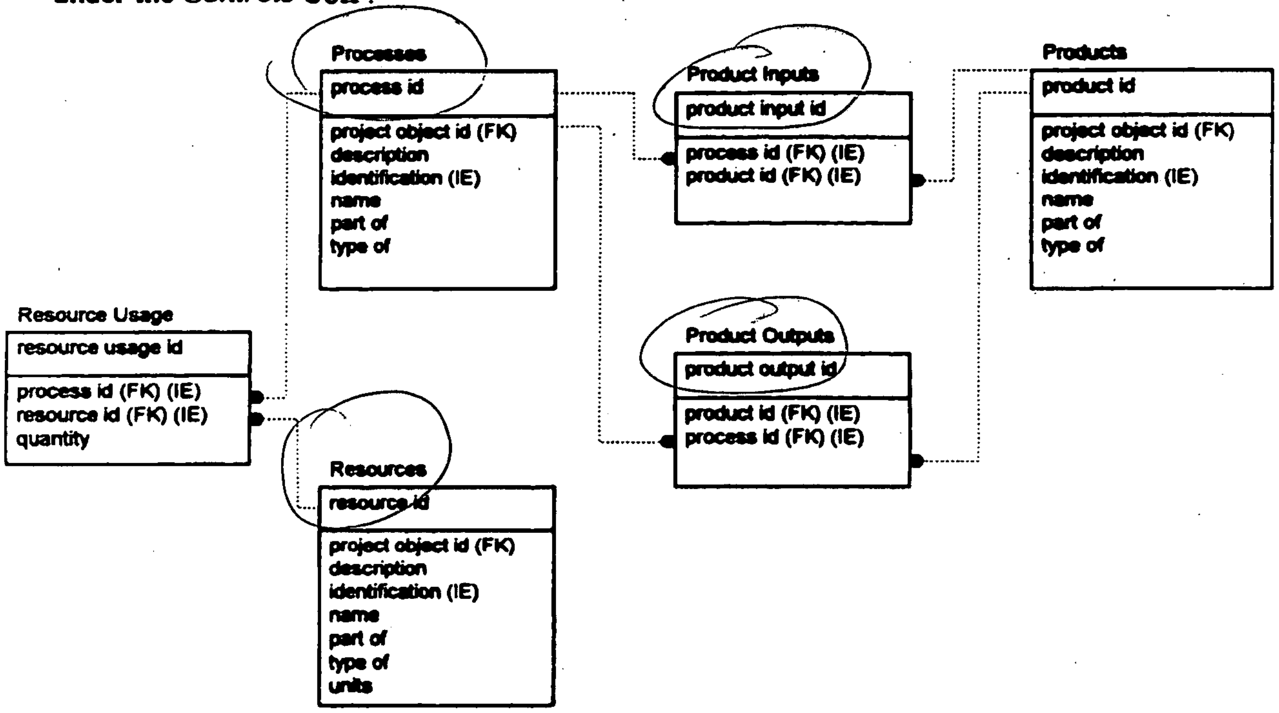


Figure 3.4: Relationships of Kernel objects IDEF1X entity relationship diagram.

3.4.2 Products

Products (Figure 3.5) are major or minor pieces of equipment or components of the project about which information is recorded (e.g., warranty information, etc.). Information is recorded for major and minor components and collections of components (systems). The product representation is not expected to be highly detailed in comparison to a model that would be required for applications such as design analysis. Components are distinguished by their origin, whether manufactured (e.g., equipment and fixtures), assembled (e.g., doors and windows), or simply space. Components can play a role in one or more systems, for example, a wall can be part of both the separation system and structural system with distinctive attributes for each role that it plays, apart from the standard attributes associated with a component.

The results of a *Process*, *Products* can also serve as inputs to a process which leads to the creation of an aggregate product, therefore this includes what are sometimes referred to as non-consumable resources. They are distinguished by their location and role in the project assembly. Products can also be acted upon by processes, in this case they are having their state changed towards the ultimate project result.

Products can be aggregated to form more complex products (e.g., sub-components form components, components form a system, or subsystems form a system). They can also have specialization relationships where a product type is instantiated with defining attributes.

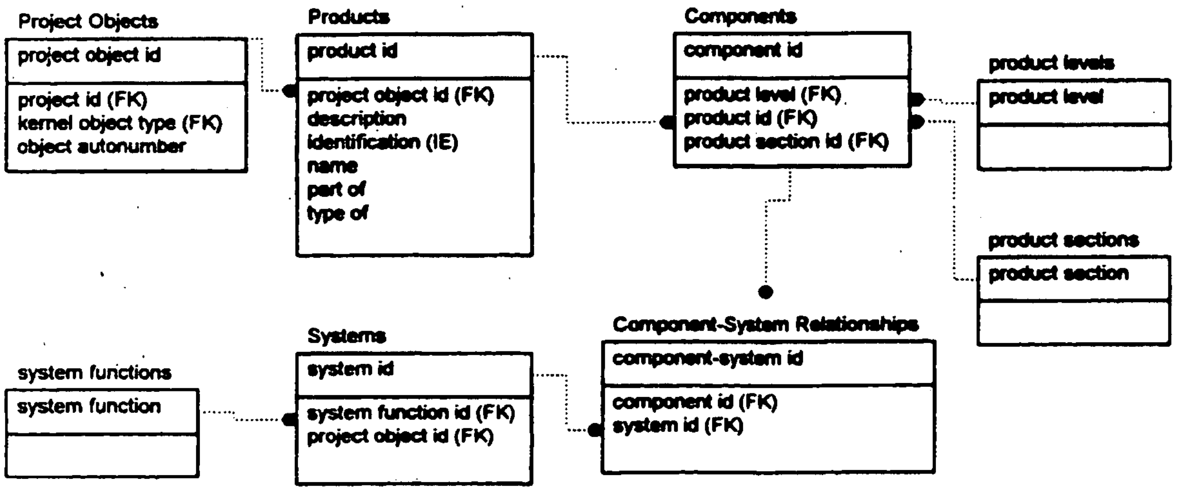


Figure 3.5: Product objects IDEF1X entity relationship diagram.

For CACP, systems are categorized by a default list, corresponding to a product breakdown structure (PBS). All systems together form a completed project. For location of products, the project can be thought of in terms of sections (e.g., buildings A, B, and C) and levels (e.g., floor 1, 2, and 3), and products are therefore located with respect to the defined project geometry.

The constructs offered by both the BCCM and the Industry Foundation Classes are more complex than is required for the PMDM. Constructs that support design related applications and represent a product's specific positioning and location within a project, as well as details concerning its geometry are not represented within the PMDM.

3.4.3 Processes

A *Process* (Figure 3.6) is an action performed towards the delivery of the project. These actions can include: construction tasks, project management tasks, procurement tasks, etc.

Processes record information about a work method, such as the type of resources involved, the types of products acted upon, the techniques used, associated cost, productivity, safety information, etc.

Aggregation of process objects results in summary tasks with simple aggregation constructs for resources, products and cost but not for time which is dependent on the logical relationships between processes. The proposed segregation for processes includes: work task, management task, and procurement task, defined as the project execution types. Lastly, processes can be constrained in the order that they are executed, this is commonly referred to as activity or process sequence.

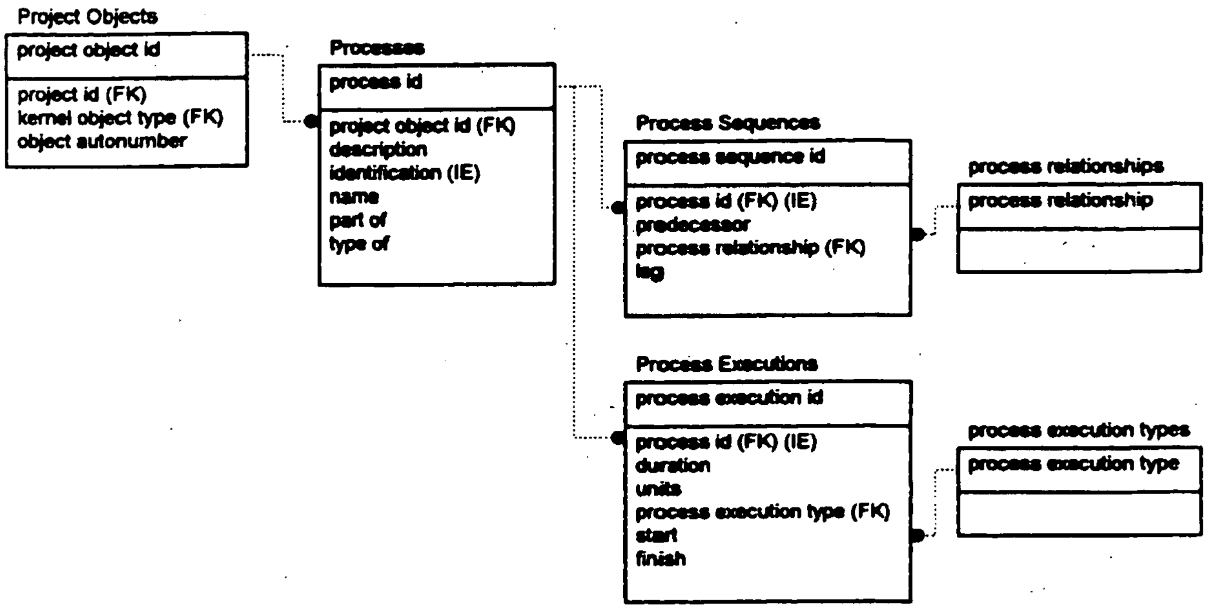


Figure 3.6: Process objects IDEF1X entity relationship diagram.

3.4.4 Resources

The aids that are required to support any process are referred to as *Resources* (Figure 3.7), and can include tools, human resources, equipment, etc. Resources can also be aggregated to form more complex resources. For example, a concrete pouring crew can be an assembly of human resources (labourers), tools (vibrating screed), and equipment (concrete truck and pump). Resource Types are also defined to refer to objects in which generic industry information can be stored. For example, the productivity for a type of human resource (carpenters) to perform a certain process (build a wall form) under a set of known project constraints.

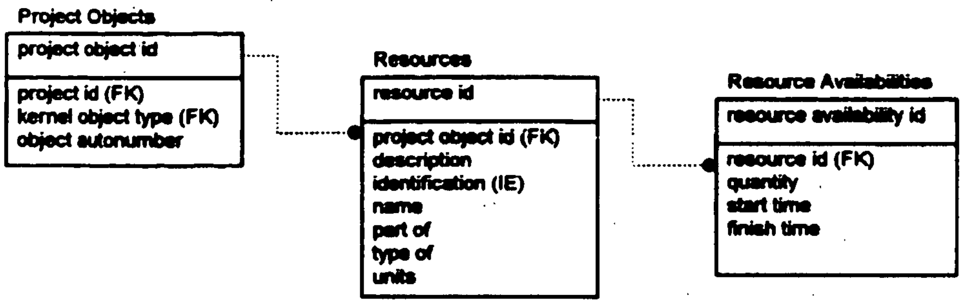


Figure 3.7: Resource objects IDEF1X entity relationship diagram.

Project resources are generally not unlimited and their availability is associated with the timeframe in which the project is executed. This availability is a constraint on the processes to be performed. The levels at which resources are represented is referred to as resource allocation.

3.4.5 Control

Requirements of, or additional project specific constraints on, products, processes, and resources, are subclasses of the kernel object *Control* (Figure 3.8). The controls are represented together as project records (e.g., drawing, schedule, estimate, etc.).

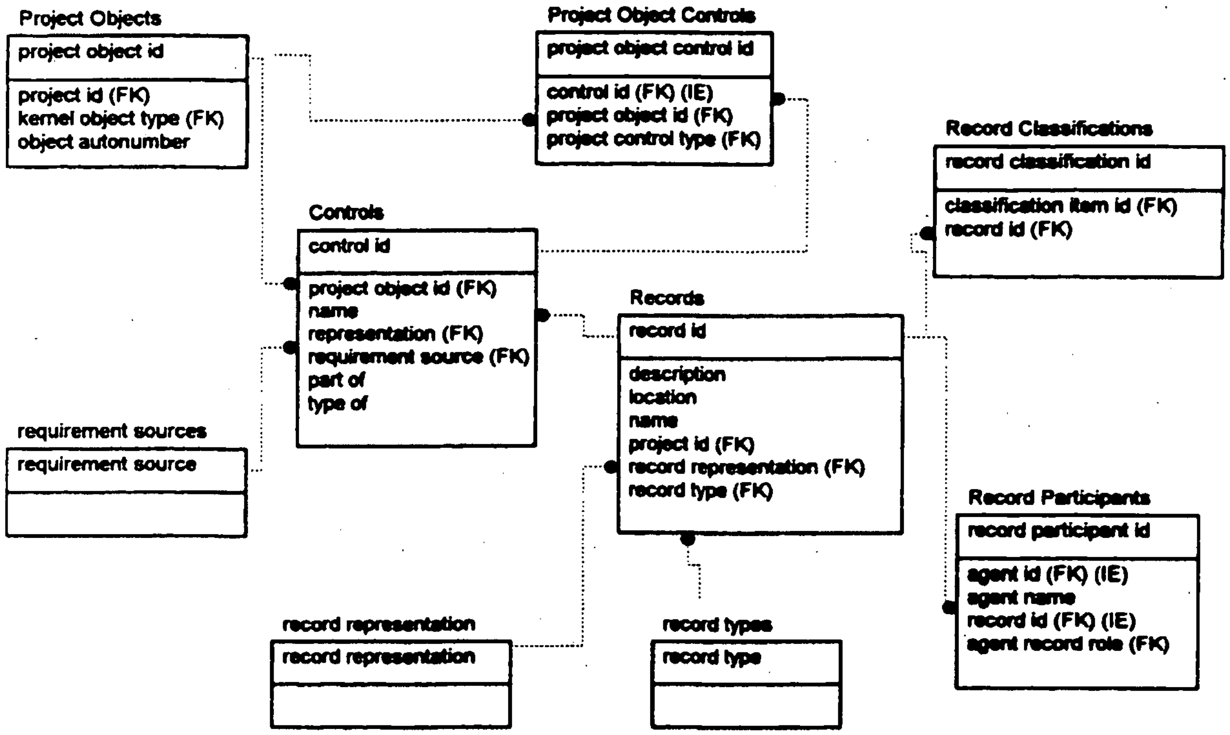


Figure 3.8: Control objects IDEF1X entity relationship diagram.

The types of project records are explicitly represented for CACP as control documents (e.g., drawings, specifications, estimates, schedules, methods, etc.). Subsections of records and relationships between records are not modeled but are indirectly related by classifying records.

The definition of *Controls* as imposing a constraint on a project object is parallel with the Industry Foundation Classes definition with the exception that the Industry Foundation Classes restrict this relationship to *Products* and *Processes*. The fact that each *Record* is a representation of a *Control* seems logical although the Industry Foundation Classes construct *Document* does not does follow this structure.

3.4.6 Cost Information

As with the BCCM and Industry Foundation Classes, costs are modeled explicitly and therefore represented as objects (Figure 3.9). There are two categories of costs, item and unit, and types of costs indicate the state of the cost (estimated, budgeted, actual, etc.), an attribute value. Costs are normally thought of as an attribute of a kernel object (i.e., product, process, or resource). For example, a product (pump) has a cost, the process (installation of the pump) has a cost, and the resources (labour and equipment required for installation of the pump) also have costs.

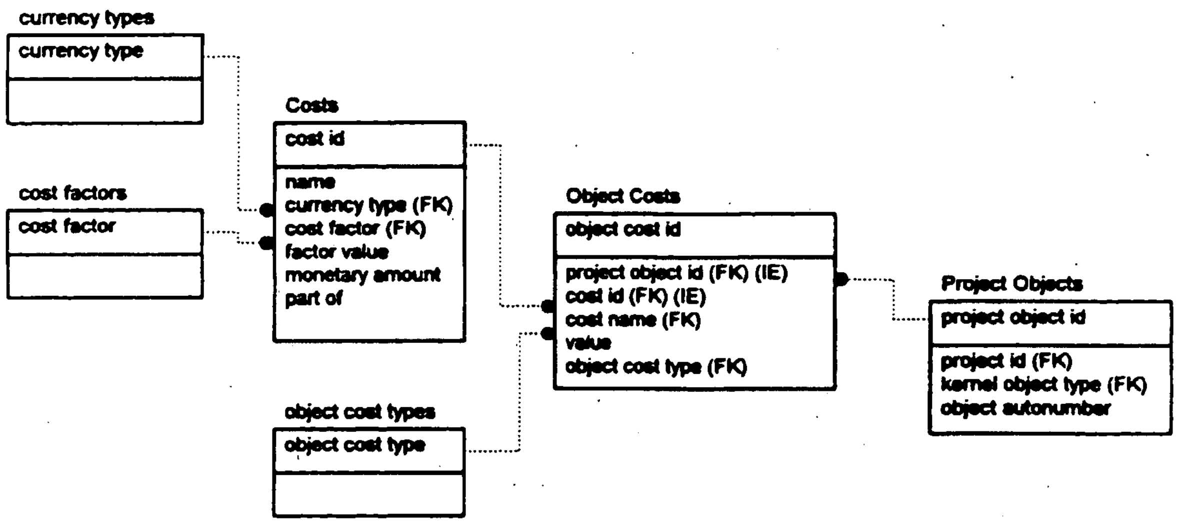


Figure 3.9: Cost objects IDEF1X entity relationship diagram.

The PMDM also recognizes that it is possible to classify the costs associated with each product by generic types that identify the processes and resources applied on a higher level. For example, a simple set of types (default and user configurable) such as delivery, installation, and operation, in addition to the cost of the product alone can be used to capture the required cost information. A draw-back to this approach is that there are cases in which costs cannot be directly attributable to a product object, e.g., temporary works.

An alternate solution is to set up classification schemes for tracking costs. These classification schemes will each invariably be structured according to a product, process and resource classification hierarchy. So, for example, a classification may identify the subcontractor direct work for building 1, system A, sub-system B, sub-item C, task 1, with resources A, B, and C. This classification item will have associated cost objects identifying estimated, budgeted, and actual values. With this approach, it will be possible to aggregate or otherwise manipulate and view costs based on a given classification.

In the cases where cost information appears on a specific record (e.g., purchase orders, contracts, etc.), alternatives include recording the breakdown of costs based on the classification scheme or simply as a cost object following the one-to-one relationship between the record and classification item. For example, a purchase order may include the supply and installation of a pump; the option exists to record the costs for supply and installation separately or together as one cost depending on the classification scheme.

3.4.7 Classifications

The representation of classifications must be flexible enough to allow the use of multiple classification schemes. For example, information can be classified according to schemes such as approaches using Masterformat (traditional and work related scheme) and alternatively Uniformat (emerging product based schemes linked to performance specifications and its approach).

A breakdown structure is a collection of specific items or category titles, usually hierarchical or nested, and numbered according to a specific coding scheme. This is a form of classification, and it is modeled based on a generalized classification mechanism (as in STEP BCCM and IAI IFCs). This approach also allows users to add their own arbitrary breakdown structures. For example, budget accounts, schedule activity codes, bar code tags, etc.

One "cost type" classification scheme might be: L - Labor costs, M - Material costs, E - Equipment costs, and S - Subcontractor costs. Here, the whole table is the "classification scheme," each line is a "classification item," the single letter identifier is the "item code," and there is only one "code segment" (i.e., a single letter).

Another "WBS" classification scheme might include: 9702-1519510010-0601-M Supply Brass Valve #601. Again, the entire table is the "classification scheme," each line is a classification item," the long identifier is the "item code". There are four "code segments" separated by hyphens, the first identifies the project number, the second gives a work type identifier (e.g., CSI-based), the third gives a unique identifier (i.e., which brass valve), and the fourth gives a cost type (M as above). In this type of a classification scheme, the user can select the project code, work type code, or cost type code from a list of allowable codes. These lists essentially make up other classification schemes (often referred to as tables). Here, for example, the cost-type code can be selected from the above cost type classification table. It might not be as simple as relating one segment to another table, however. The structure appears as that displayed in Figure 3.10.

The concept of classifying project objects is recognized in each modeling effort examined with the same basic construct represented in the PMDM.

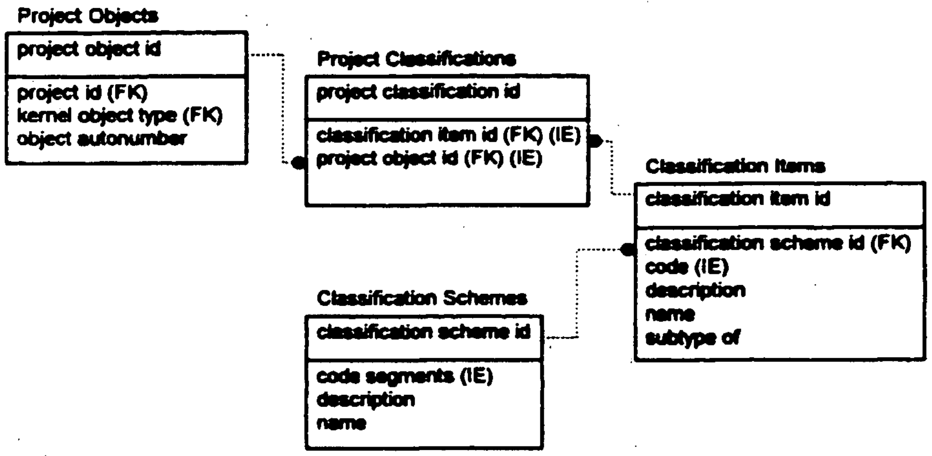


Figure 3.10: Classification objects IDEF1X entity relationship diagram.

3.4.8 Agents

There is a requirement to represent the project participants and the various roles that they play for both processes and controls (records). In addition to the roles participants play, their relationships (both contractual and communicative), as well as their individual responsibilities must be represented.

People, companies, and other organizations are referred to throughout a project, such as suppliers of equipment, parties to contracts, etc. An abstract superclass *Agents* (Figure 3.11) is included for representing relationships to *person* or *organization* data types.

The roles that people and organizations play with respect to project information are also represented. For example, a product may have a user configurable list of roles (e.g., manufacturer, supplier, installer, inspector, etc) for which an agent is identified. Alternatively, a drawing list may contain the originating company, the draftsman, etc. These lists can be constrained to require specific types of roles for certain project information in the system implementation by default (e.g., a construction process may be restricted to agents with special knowledge of its execution).

The agents in the project assume responsibilities for the execution of project processes. The responsibilities are generally defined at a higher level rather than for each individual process (e.g., all testing rather than testing for concrete).

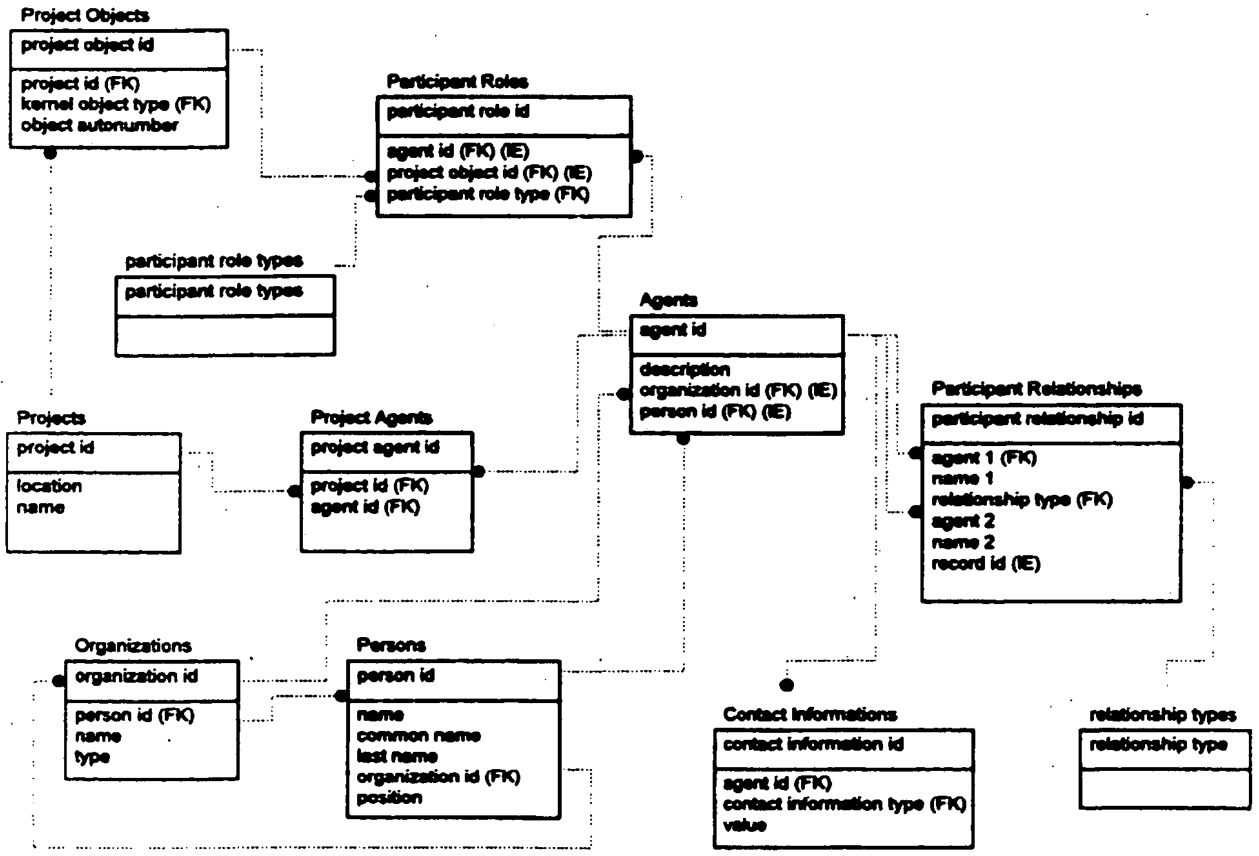


Figure 3.11: Agent objects IDEF1X entity relationship diagram.

Agents in the PMDM are related to *Project Objects*, as is the case in the BCCM, but not in the core layer of the Industry Foundation Classes where *Agents* (in the form of actors) are associated with *Processes* only. This expansion is necessary in order to capture the organizational information described above which is related to all *Project Objects*.

3.4.9 Records

As noted previously, the representation of *Controls* is most often in the form of *Records* (Figure 3.12). For CACP, project documents are subtypes of *Records*. The subtypes capture any additional attributes required for each category of the explicitly defined planning documents.

Capturing conformance to the project documents is not necessary for CACP. The records that capture the conformance to the project requirements (specifications, regulations, standards, codes of practice, see Project Requirements UofF), and include results of testing and inspection activities, as well as product certifications and process permits are not within the scope of CACP, although their templates could be.

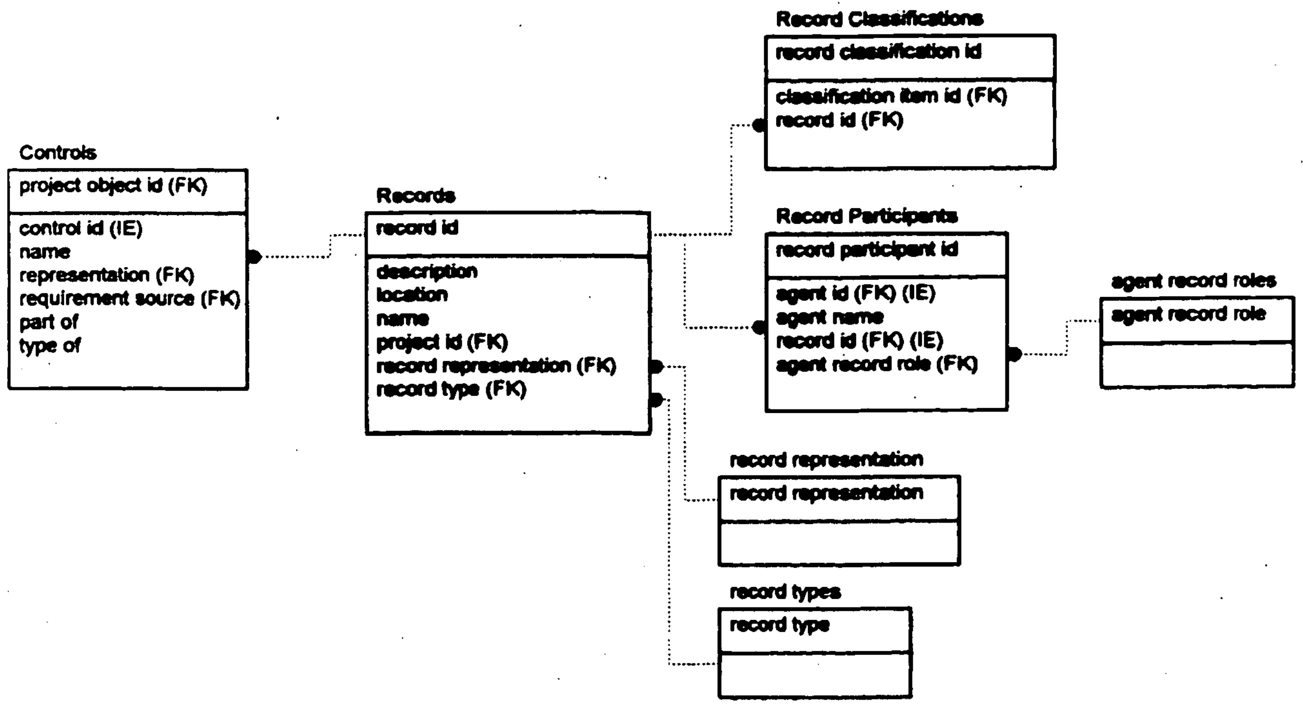


Figure 3.12: Record objects IDEF1X entity relationship diagram.

3.4.10 Constraints

An essential feature of CACP is its ability to facilitate the selection of stored past construction planning knowledge from libraries of planning information. In order to evaluate this stored information there must be some definition of the existing constraints against which the suitability of stored types will be applied.

Constraints (Figure 3.13) are part of a control's definition under which project objects exist for a given project. Constraints are used to restrict the alternatives for selection in the case of all planning perspectives (time, cost, performance, etc.). Initially, constraints are derived from an owner's objectives (e.g., a maximum cost constraint for completion of the project, a minimum area constraint on the working space of a building, etc.). Additional constraints can originate from all planning perspectives and are added and modified as a plan is developed.

Constraints are grouped according to their originating planning perspective (i.e., cost, performance, scope, time, and organization). A constraint is part of a set of constraints that limit the values for a given project object attribute. For example, a control that is represented as a specification record will carry with it a set of constraints and associated values as applied to a project object. In the case of a specification for concrete (a record representation of a performance control) it may have constraints and constraint values associated with product objects (e.g., minimum strength requirements) or process objects (e.g., minimum curing time).

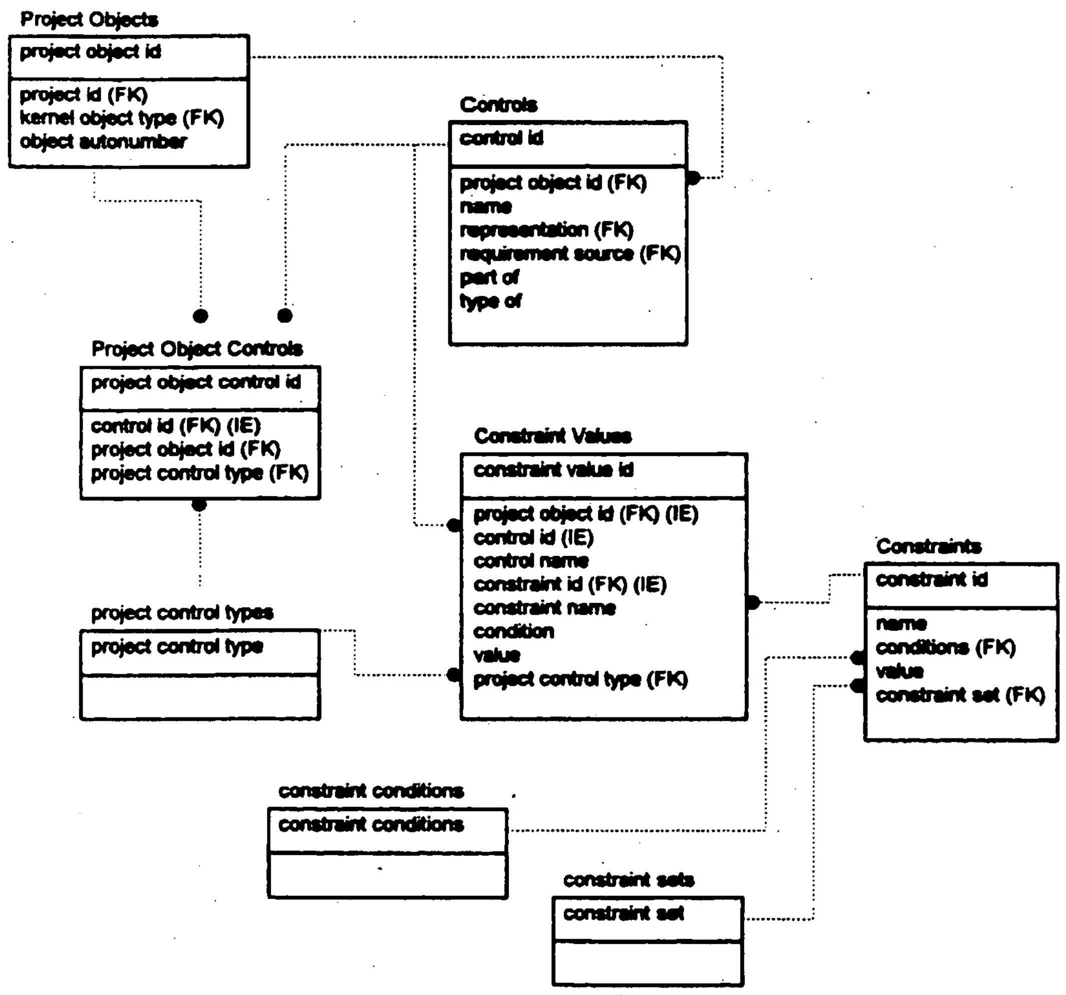


Figure 3.13: Constraint objects IDEF1X entity relationship diagram.

3.4.11 Attribute sets

For CACP, *Attribute Sets* (Figure 3.14) are required primarily for the grouping of user defined attributes for specific applications.

Attribute sets are employed in two ways:

1. They are used to add attributes that are shared by all occurrences of a given type. The *type* object has certain attributes and values assigned to it. Because the individual component is a member of a certain type, it inherits the attributes and values of that type.

2. They are used to add attributes that are different for each occurrence of a given type. In this case, the type object defines the set of attribute names that apply to each instance object, but the values depend on the specific instance.

As an example, consider the representation of doors in a large building. There may be thousands of doors in the building, but they all fall into one of about a dozen different types. Using attributes sets, all doors can be represented as instances of a general "building components" class, which defines generic attributes such as an ID code, a description, a type reference, etc. These objects are associated with an instance of component type class representing the different types of doors. Attributes that describe the type of door hardware arrangements, the glazing arrangements, the installation and maintenance requirements, etc. are all contained in an attribute set associated with the door type object. Attributes that describe the color, the key security code, the room location, etc. are all in an attribute set associated with the individual door instances.

Attribute Sets provide for run-time specification of part of an object's data structure. The main idea is that, in modeling AEC projects, particularly the facilities themselves, there are thousands of different types of elements that are candidates to be represented as distinct objects. In many cases, however, only the attributes vary from one type of element to another, not the methods or relationships. These elements can be modeled with many fewer classes (i.e., many fewer tables in a relational system), by making them all instances of a fairly general "component" class which are assigned a specific "component-type" designation and are associated with a set of attribute values (i.e., the attributes are associated with the instance, not contained in it). Not only does this lead to many fewer classes, but it allows attribute sets to be attached and detached throughout the life span of the entity, so that not all of the information generated during one phase of the project has to be "carried along" to other stages where it isn't needed.

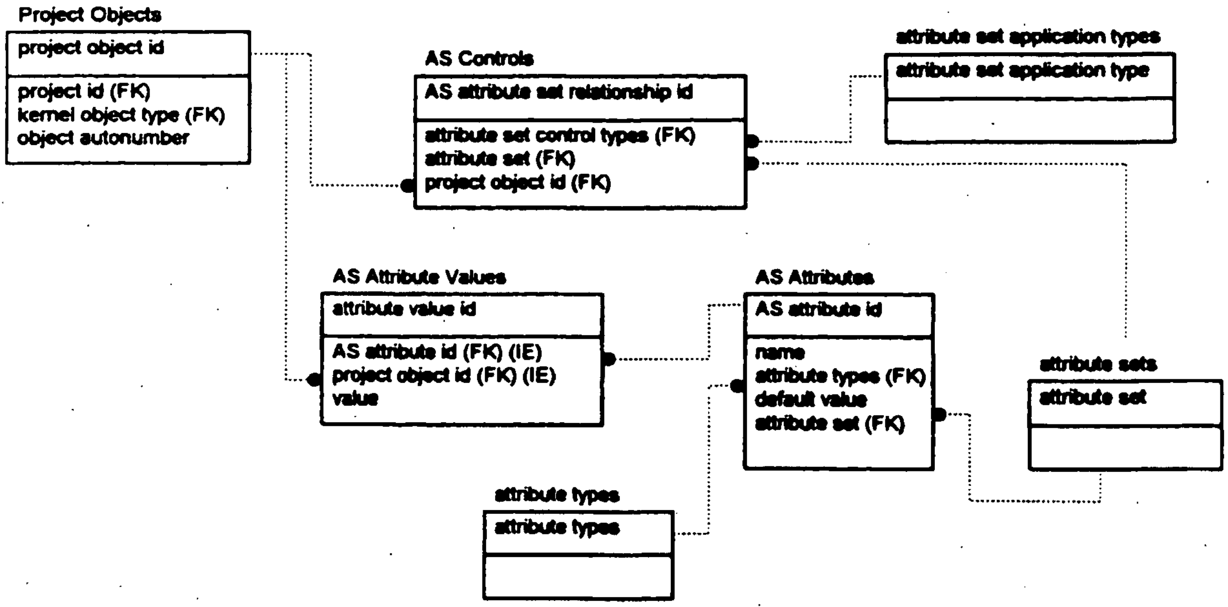


Figure 3.14: Attribute Set objects IDEF1X entity relationship diagram.

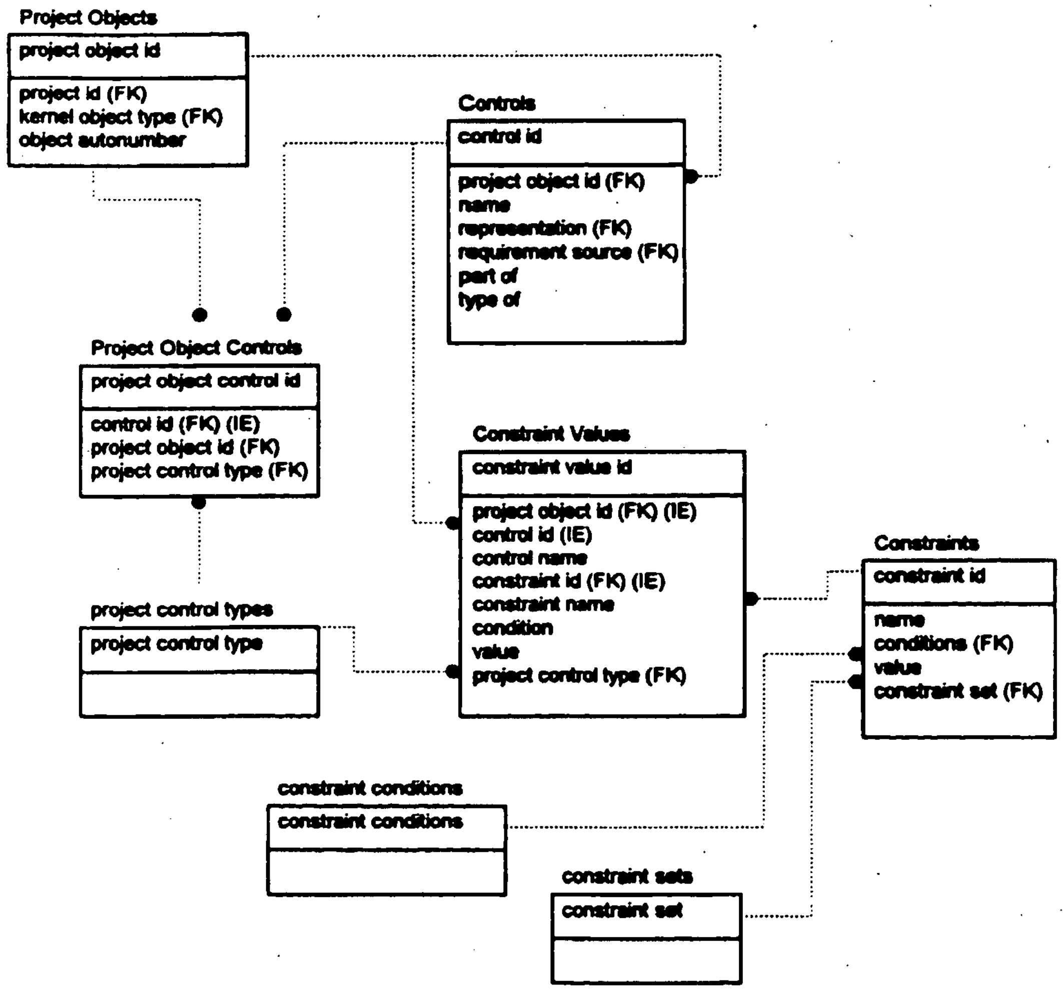


Figure 3.13: Constraint objects IDEF1X entity relationship diagram.

3.4.11 Attribute sets

For CACP, *Attribute Sets* (Figure 3.14) are required primarily for the grouping of user defined attributes for specific applications.

Attribute sets are employed in two ways:

1. They are used to add attributes that are shared by all occurrences of a given type. The *type* object has certain attributes and values assigned to it. Because the individual component is a member of a certain type, it inherits the attributes and values of that type.

2. They are used to add attributes that are different for each occurrence of a given type. In this case, the type object defines the set of attribute names that apply to each instance object, but the values depend on the specific instance.

As an example, consider the representation of doors in a large building. There may be thousands of doors in the building, but they all fall into one of about a dozen different types. Using attributes sets, all doors can be represented as instances of a general "building components" class, which defines generic attributes such as an ID code, a description, a type reference, etc. These objects are associated with an instance of component type class representing the different types of doors. Attributes that describe the type of door hardware arrangements, the glazing arrangements, the installation and maintenance requirements, etc. are all contained in an attribute set associated with the door type object. Attributes that describe the color, the key security code, the room location, etc. are all in an attribute set associated with the individual door instances.

Attribute Sets provide for run-time specification of part of an object's data structure. The main idea is that, in modeling AEC projects, particularly the facilities themselves, there are thousands of different types of elements that are candidates to be represented as distinct objects. In many cases, however, only the attributes vary from one type of element to another, not the methods or relationships. These elements can be modeled with many fewer classes (i.e., many fewer tables in a relational system), by making them all instances of a fairly general "component" class which are assigned a specific "component-type" designation and are associated with a set of attribute values (i.e., the attributes are associated with the instance, not contained in it). Not only does this lead to many fewer classes, but it allows attribute sets to be attached and detached throughout the life span of the entity, so that not all of the information generated during one phase of the project has to be "carried along" to other stages where it isn't needed.

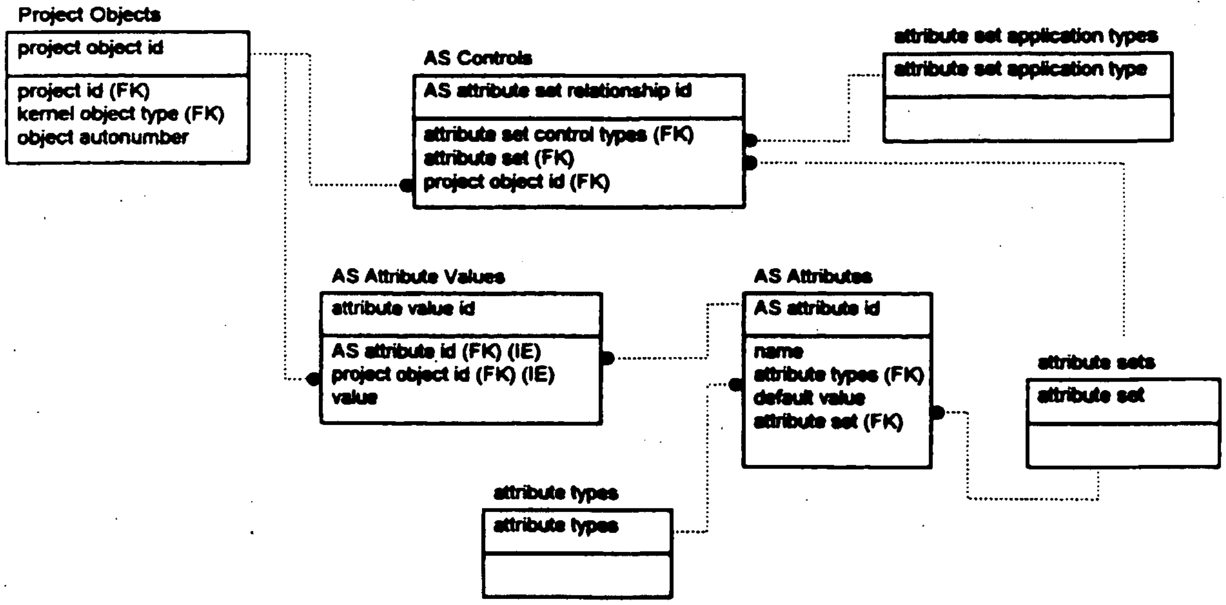


Figure 3.14: Attribute Set objects IDEF1X entity relationship diagram.

The concept of attribute sets is also found in IAI's Industry Foundation Classes. The Industry Foundation Classes employ attribute sets in the two ways described above (attributes and values, and attributes only) and in addition they are provided as a mechanism to allow future growth and flexibility by allowing end-users to extend the data model through the addition of customized attribute set definitions.

3.5. Chapter Summary

In general, the PMDM has simplified previous modeling efforts examined in its representation of *Products*, as the level of detail required for the representation of product information is not as extensive for project management functions as it is for other applications within the AEC. The PMDM has also elaborated on and modified the areas of *Agents*, *Attribute Sets*, *Constraints*, *Controls*, and *Records* in order to fulfill the scope defined in the process and functional analysis for construction planning. The distinction between adopted objects and newly developed objects is provided in Appendix A.

Chapter 4: Computer-Assisted Construction Planning

The focus of CACP is the storage and reuse of assemblies of planning knowledge. As noted earlier, previous efforts towards automating the planning process have centered on low-level reasoning about project components to generate a time-scaled activity sequence. More recent efforts have emphasized broader information technology approaches rather than tightly focused artificial intelligence solutions. These approaches have a higher level of information representation and use templates or libraries of knowledge structures and are more closely related to the approach taken for CACP.

4.1. Conceptualizing CACP

CACP builds on this previous work in generating construction plans and as a result of being based on an integrated construction management approach extends it in the following areas: 1) broadens the scope of planning beyond project scope, time and cost, 2) supports integration with other PM tools by relating the planning directly to standard project data models, and 3) treating the capture and reuse of past project knowledge as a technique of managing library information.

4.1.1 CACP Features

CACP supports integrated systems by supplying the initial information required in a planning mode. Rankin et al. (1998) describes the functionality of CACP and describes its key features as:

1. Data modeling - is based on an integrated data model with its roots in a general integrated project management data model.
2. Type and part information storage structures - allow efficient storage and retrieval, support multiple levels of detail during development through representation of general information (libraries), and specific information (current project).
3. Case-based reasoning - employs artificial intelligence methodology for initial plan development, CBR is presented as a methodology rather than a technology (the assisted component).
4. Rapid application development - tools are used for its development in consideration of integration and ease of implementation (low cost for a low cost technology industry) while also providing sufficient functionality.
5. Interfaces and classification - follow a growing trend in representing information that is hierarchical in nature by using an effective 2-dimensional representation of information in a "tree and detail" combination.

The first item was discussed in the previous chapter. This chapter covers the second and third items and establishes the scope of construction planning in the context of CACP while explaining the planning approach adopted to guide the development of the prototype system. The remaining items are covered in the next chapter.

4.1.2 CACP General Architecture

CACP can be thought of as one of many tools available within a toolbox of integrated project management applications. It is however, a key tool for giving the user a "head start" in many planning tasks. The architecture of CACP is presented as consisting of three parts: the Database, the Interface, and the Reasoner. Figure 4.1 depicts each of these parts in the context of *scope planning* and its required functions (Chapter 3).

The Interface allows access to project information in a variety of views and levels of detail, acts as an interface to data sources and allows consultation with a reasoner for the development of a plan. Knowledge is stored in type libraries and structured according to the PMDM (with the addition of a standard classification scheme) and aggregated in planning templates.

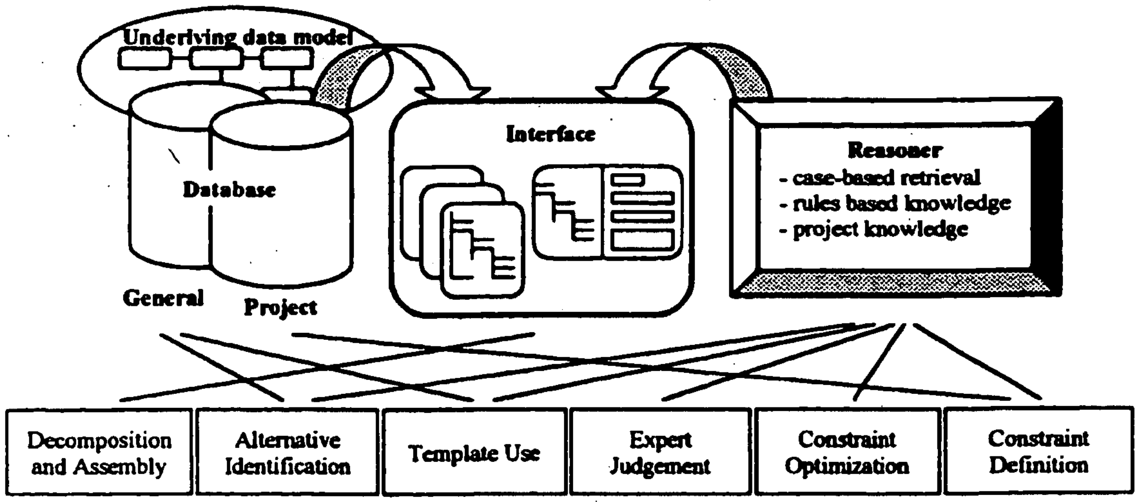


Figure 4.1: Architecture of CACP with scope planning techniques.

All project information is stored in a common database as structured according to the PMDM. The applications used to further modify the plan developed, which are outside the scope of CACP, include traditional scheduling and estimating components, as well as support for elaboration on planning information such as detailed methods statements and organizational structure for a given project.

4.1.3 Computer Assisted vs. Automated

Many researchers have adopted an approach to assist users in the development of construction plans by re-applying past planning knowledge in the form of "templates" (Aalami and Fischer 1996, Chevallier and Russell 1998, Jägbeck 1998, Warszawski 1995, Yu 1994). As suggested earlier, this trend can be partially attributed to the lack of success that previous attempts have had in automating the development of practical construction plans based on low level planning information. CACP also subscribes to an approach of capturing and reapplying templates of planning knowledge to assist the user in development of a construction plan rather than attempting to automate this complex problem domain. This approach also remains consistent with a management philosophy that promotes "planning" as an essential task in the performance of all construction work.

The complete automation of the planning process would skip an important step in the proper execution of construction tasks by robbing the planner the opportunity to initially build the project "in their mind" prior to execution in the field.

As a precursor to the development of the reasoning component of this research, simple applications of capturing planning knowledge were examined by the author through a project initiated by the New Brunswick Department of Transportation (NBDOT). The project entailed the development of an information system to support a design engineer's planning of in-house design projects. These design projects begin with the initial assignment to the responsible design engineer and conclude with the tender of the design for construction, and they involve the interaction and co-ordination of many different branches within the department.

Planning templates were developed for a few typical design jobs (highway and bridge design projects). The templates have been implemented in Microsoft's Project 98, primarily due to the NBDOT's commitment to implement other Microsoft products, and the integration and communication advantages that this environment provides. In the simplest scenario, knowledge of the existing template is captured as either part of the database *note* field associated with each task or as a "hyperlink" to any file type and location (e.g., a URL). This approach is similar to that used in some of the IS tools discussed in Chapter 2. The results of the project reinforced a few of the advantages of this approach including an ability to plan faster with an increased level of accuracy and the transfer of planning knowledge through its capture in templates.

4.2. The Planning Approach

The approach for the development of a construction plan based on the information structures proposed for TOPS (and hence CACP) is laid out in a paper by Froese et al. (1997a) and further described in Froese and Rankin (1998a). The following subsections elaborate on the principles presented in these papers and relate them to CACP application.

4.2.1 Modeling Concepts

The approach is described in terms of the PMDM kernel objects and the basic data modeling concepts of: *type of* versus *part of*, *aggregation* versus *specialization*, and *type* versus *instance*.

Information represented in terms of the kernel objects (products, processes, resources, and controls) can be organized in a hierarchy of increasing detail. The hierarchies contain aggregation constructs where objects are made up of sub-parts (Figure 4.2) or specialization hierarchies where objects are generalizations of more specific sub-types (Figure 4.3).

Both aggregation and specialization hierarchies are useful to apply to a specific type of object but only aggregation hierarchies are applied to instance information. For example, it is meaningful to speak of sub-types of a building's foundation (e.g., pile foundation, strip foundation, raft foundation). It is not as meaningful to define sub-types of the instance *building pile foundation*, however sub-parts for an instance (e.g., pile #1, pile cap #1, etc.) are defined.

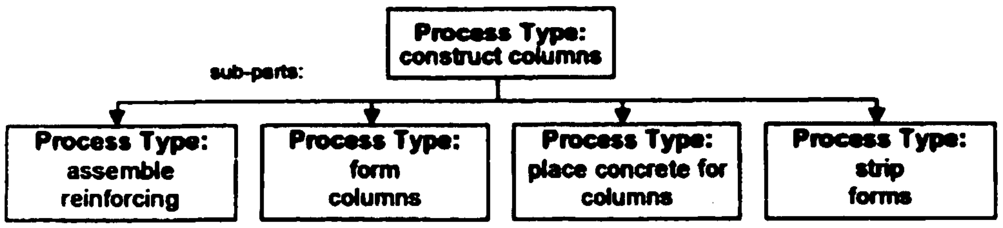

sub-parts:
Process Type: construct columns
Process Type: assemble reinforcing
Process Type: form columns
Process Type: place concrete for columns
Process Type: strip forms

Figure 4.2: An aggregation hierarchy of process sub-parts.

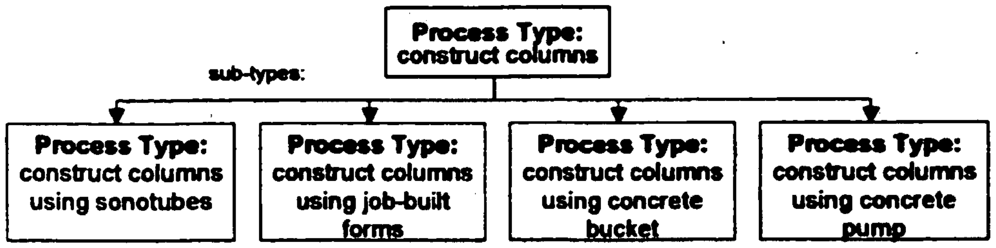

sub-types:
Process Type: construct columns
Process Type: construct columns using sonotubes
Process Type: construct columns using job-built forms
Process Type: construct columns using concrete bucket
Process Type: construct columns using concrete pump

Figure 4.3: A specialization hierarchy of process sub-types.

As depicted in Figure 4.4, the "part" and "type" hierarchical relationships for a single entity (e.g., products, processes, or resources) and the fundamental relationships between two entities (e.g., a process *applies to* a product), are used to developed a plan. This is accomplished by selecting from libraries of stored type information structured according to the PMDM.

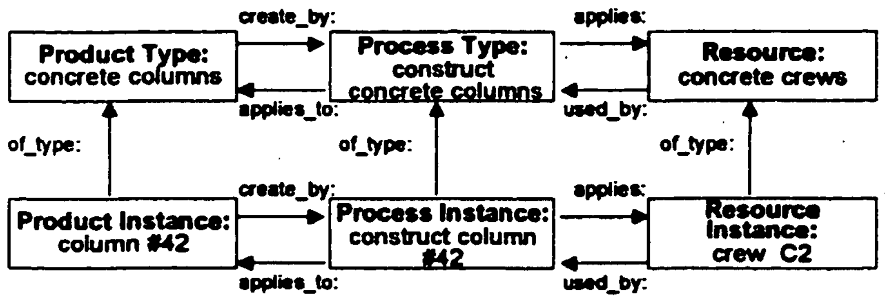

create_by:
applies:
Product Type: concrete columns
Process Type: construct concrete columns
Resource: concrete crews
applies_to:
used_by:
of_type:
of_type:
of_type:
create_by:
applies:
Product Instance: column #42
Process Instance: construct column #42
Resource Instance: crew C2
applies_to:
used_by:

Figure 4.4: Fundamental relationships for kernel objects.

In the CACP approach, different levels of detail are not explicitly modeled. For example, different entities to represent processes at the level of overall projects, project phases, work packages, activities, tasks, etc. are not defined. Rather, all of these levels can be represented using the basic process entity and an aggregation relationship to define the linkages between the levels.

4.2.2 The Methods Approach

A common theme to research surrounding the application of templates to construction planning is the concept of "methods" as a means of storing assemblies of planning knowledge. The term "methods" has been adopted by many to define some assembly of products, processes, and resources. An earlier system, the Advanced Construction

Technology System (Ioannou and Yiu 1993) recorded information about construction methods in a database but mainly in the form of unstructured text. Recent research has recognized the importance of explicitly representing "methods" or "types" of planning assemblies.

Chevallier and Russell (1998) defined "activity fragnets" as capturing the construction activities required to construct part of a physical component breakdown structure. The fragnets include precedence relationships and activity logic and serve as "type" definitions that are instantiated by applying refinement rules relating to the scale of application. Fischer and Aalami (1996) have taken the approach of capturing "type" information in a method model consisting of the constituting activities (with sequence), and their resource requirements, and the products they can be applied to. In this case, selection of a method is guided by a set of rules that define applicable methods for a given activity and product, thereby yielding lower level activities. Jägbeck (1998) also defines "methods" as a "type" structure that may be applied to a given product to form a "plan". In MDA Planner (Jägbeck 1994) a method has preconditions that required satisfaction before it could be applied to a product.

CACP builds upon the underlying core model in the representation of methods. The ideas of process types and individual processes as representations of construction work performed have been introduced. A construction method entity would represent the way in which a process is carried out. For the purpose of representing and managing construction methods, no distinction is introduced between terms such as methods, work techniques, construction technologies, etc., and thus all of these terms are treated as synonyms. Also a distinction between method types and individual instances of methods does not seem useful here since there is no notion of a specific instance of a method that can only exist one time.

The information required to describe a process type includes the following attributes: the type of product being acted upon, the type of participants carrying out the process, productivity and unit cost rates for the process, types of predecessor and successor processes, types of locations where the process will occur, types of resources used by the process, and the work techniques and arrangements used by the process.

The information required to describe the corresponding work method—work techniques, arrangements, resources, etc.—is a subset of this same information. The attributes for an individual process would be similar to process types, but could identify instances rather than types of the related information. There seems to be a one-to-one relationship between a method and a process type, since for each distinct work method that can be defined, it seems useful to also define a corresponding distinct process type.

An important consequence of this approach is that, although process types and methods can be defined as two different entities, there is a one-to-one relationship between them and there may be no reason to represent them as distinct entities in a data model. That is, *methods are represented through the representation of process types.*

4.2.3 Methods in CACP

Adopting the modeling principles discussed above, the methods approach for CAPC is described. In the context of processes, for example, a process type's sub-parts are the sub-processes that must be carried out, while its sub-types are the alternative techniques for carrying out the process. Assisted planning, then, can take place using an approach of refining a high-level process by expanding the plan according to the aggregation hierarchy and selecting alternative techniques according to the specialization hierarchy.

More specifically, a system can contain descriptions of all types of building products used for a particular class of construction (see Figure 4.5). These descriptions are arranged into both aggregation hierarchies (i.e., which components are sub-parts of others) and specialization hierarchies (i.e., which components are sub-types of others). Similarly, the system contains descriptions of all types of construction processes generally used within the class of construction, again arranged in aggregation and specialization hierarchies. Relationships between process types and product types indicate which processes are appropriate for various products. Similar relationships relate processes to the resources and constraints under which they are appropriate.

As with all "methods" approaches, a description of a facility to be constructed is required, regardless of the level of detail at which it is represented. This takes the form of product instance information arranged in an aggregation hierarchy and related to the product type information. This approach to planning does not necessarily require detailed project information in advance. As in Fischer and Aalami (1996), planning starts at a high level and proceeds to whatever level of detail exists for the product description. A high-level "seed" process is adopted and becomes the current planning target (such as "construct building"). If multiple sub-types exist for the target's process type, then these represent the alternative techniques that can be used for carrying out the process. A selection must be made from among them; this can be accomplished using expert system rules or in combination with input from the user. Any known information about the type of product, resources, and project context can also be used. It is also possible to consider multiple alternatives simultaneously and defer the decision until more detail has been determined. Once a selection is made, the sub-type becomes the new planning target. If multiple sub-parts exist for the target's process type, then each one of these sub-parts must be carried out; each is added to the plan and becomes the new planning target in turn. Some sub-parts may not be necessary or appropriate in a particular situation; again decisions are made manually or automatically about including them.

In this planning approach, much of the knowledge about planning resides in the product and process type breakdowns. Additional knowledge can reside in rules used to make selections from available alternatives. This knowledge representation scheme seems relatively easy to maintain and to update with information from previous projects.

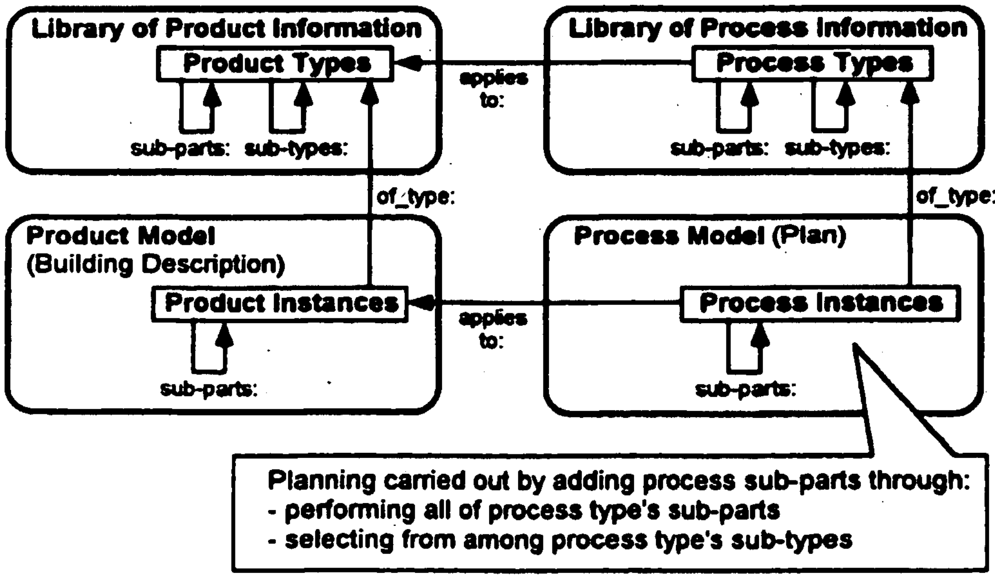

Library of Product Information
Product Types
sub-parts:
sub-types:
applies to:
Library of Process Information
Process Types
sub-parts:
sub-types:
of_type:
of_type:
Product Model (Building Description)
Product Instances
sub-parts:
applies to:
Process Model (Plan)
Process Instances
sub-parts:
Planning carried out by adding process sub-parts through:
- performing all of process type's sub-parts
- selecting from among process type's sub-types

Figure 4.5: Elements of a computer-assisted planning approach.

4.3. *Case Based Reasoning*

Instead of attempting to represent additional knowledge for alternative selection or scaling purposes in the form of hard coded rules (e.g., IF-THEN statements), CACP explores a case-based reasoning (CBR) methodology for its decision making process. The case based reasoning methodology is selected in recognition of the availability of a broader range of project information then was previously available and the constant changes in construction processes that the information represents. The case-based reasoning approach simplifies the elicitation process by looking beyond the need or maintenance of hard coded rules for the representation and selection of industry knowledge such as the approaches proposed by Aalami and Fischer (1996) and Chevallier and Russell (1998). With the many possible variations of construction technologies or methods and the possibilities of special conditions that can be introduced by each new project, the maintenance of highly structured decision trees or IF-THEN statements as solution for guiding the user through a selection process is a challenge.

Implementing case-based reasoning for CACP is not intended as advancement in case-based reasoning planning technology but rather as a technique for extending the management and application of stored project management information. Therefore, a complete implementation of the case-based reasoning approach is not attempted, but rather the basic steps are considered in order to demonstrate the concept as it applies to computer assisted construction planning. The case-based reasoning algorithm depicted in Figure 4.6 is a reproduction from Riesbeck and Schank (1989) and indicates the extent of the algorithm's implementation for CACP. The shaded boxes indicate the steps in the algorithm that have been considered within the scope of this research.

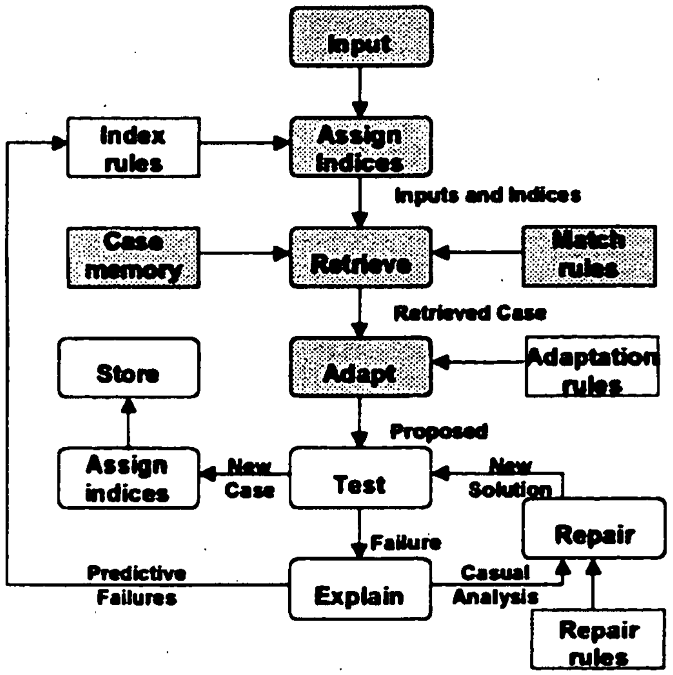


Figure 4.6: Case-based reasoning algorithm (Riesbeck and Schank, 1989).

4.3.1 Case Based Reasoning for CACP

The elements of case-based reasoning, as they relate to the implementation in CACP, are summarized in Table 4.1. The table lists the key issues involved when implementing case-based reasoning for any application and notes the solution adopted for CACP, as well as the implementation in relation to the conceptual description of CACP. As with the case-based reasoning applications to design engineering noted in Chapter 2, implementation issues primarily surround the representation of cases, in consideration of the methods of indexing for effective retrieval and adaptation.

4.3.1.1 Case Representation

While the *Representation of cases* has presented challenges to applications in other domains implementing case-based reasoning, it is lessened for CACP due to the adoption of well founded standard data modeling techniques where all project information is explicitly represented. The development of the PMDM, therefore, contributes significantly to the implementation of a case-based reasoning approach. It should be noted, however, that the current data model and its implementation is restricted to capturing simple attributes types (e.g., textual and numeric) without consideration for information of a graphical nature. As explained earlier, the representation supports a hierarchical approach following sound information technology principles and the direction established by standard industry data model efforts within the construction management domain.

Table 4.1: Case-based reasoning implementation for CACP.

CBR Option	CACP Solution	CACP Implementation
Representation of cases:		
Attribute-value pairs	• attribute-value pairs	• PMDM
Representation model	• a model that supports part and type hierarchies	• PMDM
Attribute types: text, numeric, images, & drawings	• text and numeric, not images or drawings	• implementation (MS Access relational database
Indexing:		
Partitioning	• storage based on the kernel objects	• PMDM
Clustering	• generated during a reasoner session	• reasoner component
Dynamics	• indexing dynamically generated during retrieval	• reasoner component
Retrieval:		
Context	• guided by relationship attributes	• PMDM
Nearest neighbour, induction	• similarity matching on attribute fields	• reasoner component
User control	• flexibility priorities	• CBR interface
Adaptation:		
Assistance	• guidance provided	• tree interface
Automation	• minimal	

4.3.1.2 Indexing

Properly *Indexing* information in any information management system is important for an efficient retrieval. Two perspectives on indexing are partitioning (how information is divided) and clustering (how information is grouped within the resulting partitions). The model inherently partitions information according to the kernel objects of the information stored (i.e., products, processes, resources, controls). Clustering of information is achieved through any of the attributes associated with a given object (e.g., products clustered according to a classification object, a related process object, a related control object, etc.). While partitioning is effectively performed during design of an application, clustering of information is performed by a case-based reasoning application during "run-time". The criteria for retrieving information will change as the project constraints are revealed and therefore the attributes selected for indexing are dynamic and generated during each retrieval session. Based on the attributes selected a case-based reasoning application will cluster information for search, comparison, and retrieval tasks.

4.3.1.3 Retrieval

There are basically two approaches to *retrieval* of cases in case-based reasoning: *nearest neighbour* which calculates distances between the target problem and stored cases and *inductive retrieval* which builds a decision tree based on the information in the case base and runs a target problem through the decision tree. Separately, each has its advantages over the other during implementation. Decision trees are generally quicker in their execution because the number of cases considered is reduced at each step in the tree while nearest neighbour considers all cases in the case base. However, nearest neighbour retrieval does a more efficient job with incomplete data as it can still calculate distances for the attributes of the target problem that do have values whereas with a missing value in a decision tree the retrieval process can fail. It is also possible, with some case-based reasoning tools, to combine the two techniques effectively reducing the number of cases to consider in the nearest neighbour approach by applying simple decision trees.

As an initial implementation, CACP employs nearest neighbour retrieval as it is expected that the information in a case base will be incomplete and therefore not conducive to inductive retrieval. There is, however, a certain degree of partitioning of the case base information by examining specific attribute values that are assured of being present in both the target and the case base. For example, when looking for a matching method a product type and process type will have been defined, and these attributes can be used to reduce the number of cases to consider.

4.3.1.4 Adaptation

CACP's term "Assisted" refers in part to the case-based reasoning *adaptation* step. At this point in the development of CACP, adaptation is left to the user, with minimal assistance provided by the system. In case-based reasoning terminology, this is referred to as *null adaptation*. It is not uncommon for a case-based reasoning system to not use adaptation, however, there are techniques that apply rules based on the differences between the target problem and the retrieved case and guide the adjustment of specified attributes. As implemented for CACP, alternatives are presented as a result of conducting a reasoning session, but adoption of a retrieved case—as well as determining the scope of what is adopted—is left to the user.

4.3.1.5 Case Acquisition and Weighting

The development of CACP has only explored the use of case-based reasoning for its implementation and, therefore, a complete case-base has not been developed. The acquisition of representative cases required to build-up a working case base and the issues that surround it were not within the scope of this research. Without a reliable case-base, it is also unnecessary to consider some of the techniques that can be applied for fine tuning its performance. Weighting of the attributes used for similarity matching is one such technique and, while certain allowances have been made for weighting in the application, its relevance has not been validated. Development and tuning of case-bases would be a significant step for a complete implementation of CACP. However, the groundwork has been laid through the development of the integrated data model and prototyping of the interface to facilitate this process.

4.3.2 Planning with Case-Based Reasoning

Planning with case-based reasoning in terms of the data model presented in Chapter 3 and the data model concepts (planning approach) is presented in section 4.2. The advantage of this approach is that rules for additional constraints do not have to be written. The "Reasoner" considers each alternative selection based on the attributes and constraints present in both the project specific information and general information and chooses best match(s) at the time of selection. As new attributes or constraints are added to the project information so are the conditions under which a comparison is made.

Figure 4.6 presents the conceptual representation of a number of cases for consideration in a selection process. A single case comprises project specific information that includes an instance, its type and the conditions under which it exists by representation of its relationship to controls, constraints, and other attributes (including relationships to other objects and subparts if available). A number of these project specific cases make up the case base for alternative analysis.

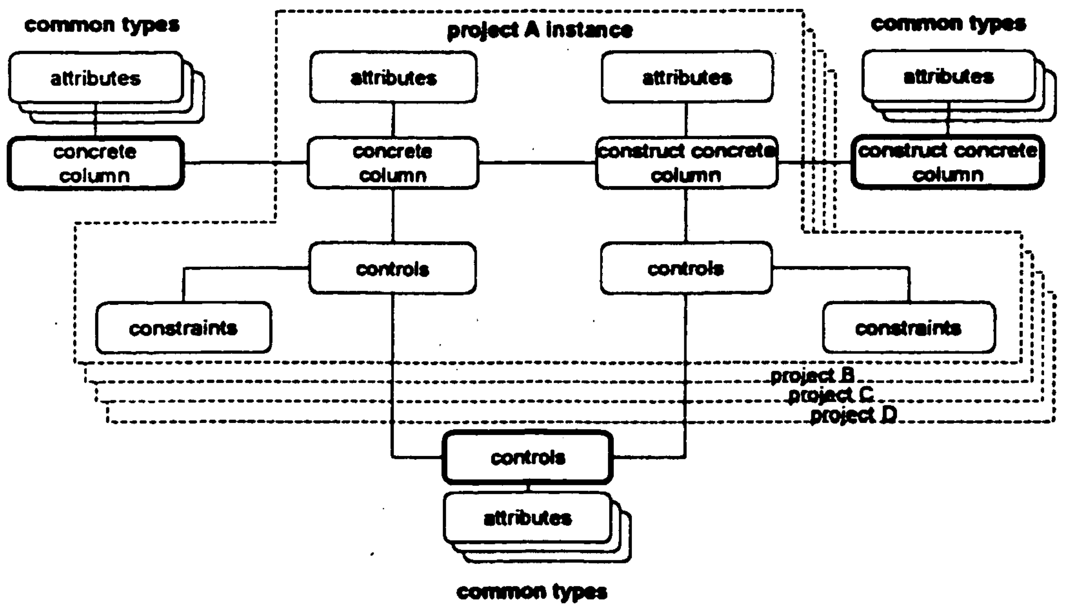


Figure 4.6: Conceptual representation of cases.

Consider Figure 4.7, which depicts the process of selecting a process instance to fulfill the relationship of *produces* "concrete column" by picking the appropriate method to "construct concrete column". The process always begins at a level of picking types. Based on a comparison of what attributes and constraints exist in the current project and those that are present for solutions to *produces* "concrete column" in the available cases, a set number of types are proposed for selection. Having selected a "type," questions regarding the related information to the "type" must be addressed: what additional constraints should be added to the existing project information, what additional related objects should be added, should instances of the object's subparts be established?

By deciding whether to add subparts and relational attributes the user is first selecting the level of detail at which a project's information is being developed and secondly the importance of additional information to the current project. The user can thus cut off the representation of information detail at any level in a hierarchical structure.

For example, if we consider picking a process "type" such as "construct the concrete column using sono tubes," the aggregation of its parts must also be given consideration, and any special conditions that apply to the case selected. Will the user choose to manage the information for this project work at the level of detail of this "construct the concrete column using sono tubes" process, or choose to treat it as an aggregation of the processes "set forms," "place rebar and inserts," "pour concrete," "cure and finish concrete," "strip forms," etc.?

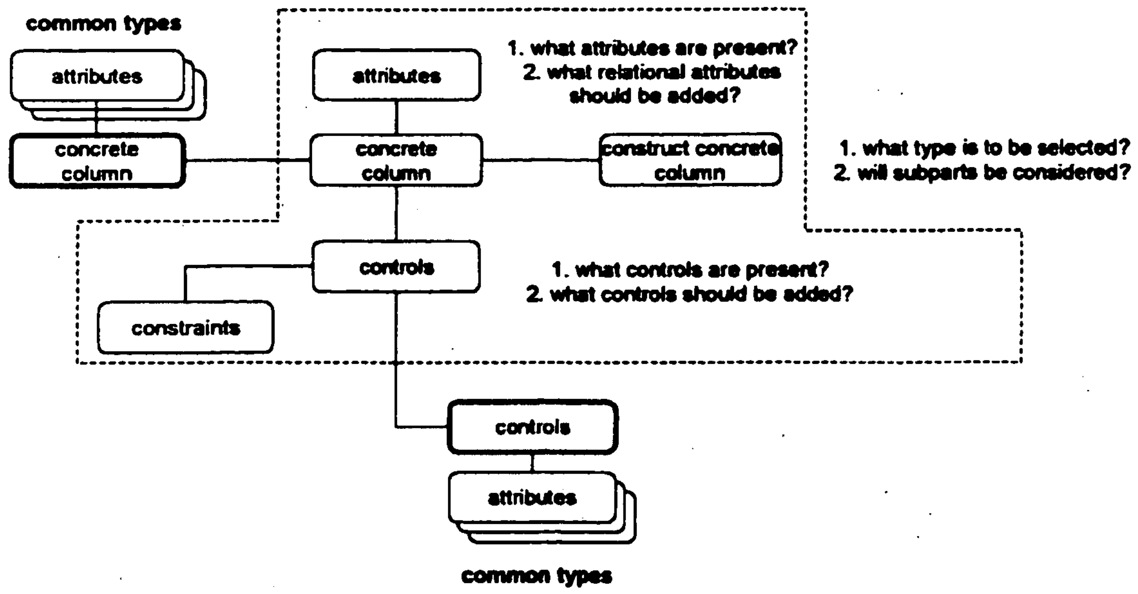

common types
attributes
concrete column
attributes
1. what attributes are present?
2. what relational attributes should be added?
concrete column
construct concrete column
1. what type is to be selected?
2. will subparts be considered?
controls
constraints
1. what controls are present?
2. what controls should be added?
controls
attributes
common types

Figure 4.7: Questions answered by case-based reasoning.

When looking to a case-base in selecting a process, consideration must also be given for the conditions under which the case instance exists. Are all the attributes applicable for this case (current project) and what additional constraints should be added to the project that are not already present in the current project information? For example, "pouring concrete for sono tubes" may have additional constraints for structural concrete with high slump due to the congestion of rebar and inserts. What other currently identified processes should these constraints apply to?

Cases will also contain relational attributes. In the case of the "construct the concrete column using sono tubes" process, relational attributes can include the resources required as part of this process. If resources of the type specified in the case are not present in the current plan then the user has the option of adding this information as well. For example, the process case may include the resource of type "concrete pump" which was not previously present in the project information, the user then has the option of adding an instance of this entity, thus further populating the current database.

Answering the questions noted above completes the adaptation step in the case-based reasoning approach. Again, the adaptation process is not considered automated but rather "supported" by the application.

4.3.3 Planning Granularity

The level of detail at which a user decides to work will be dependent on the complexity of the work being planned and should therefore be consistent with the expected level of detail at which a project will be controlled. This level of detail will vary from one organization to the next and certainly from one type of work to the next. The level of detail represented in this research was selected in order to explain the functionality of the tool and should not be taken to indicate the required level of detail at which a user must work. Working with information at a finer level of detail focuses the discussion on the process of planning rather than the content of the planning information. Therefore, the level of detail described in the example is independent from what could be used in an actual implementation.

In consideration of the appropriate level of detail at which to work, as a starting point, the libraries of type information for each kernel object provide a structure based on standard industry classifications of construction information. However, they are not restricted to a set number of hierarchical levels. It is then up to each individual user to decide the appropriate level of detail at which to work. As a result, the level of detail available for the reasoning steps is dictated by the availability of past cases. As cases are built-up (e.g., as current project databases are saved to act as cases for future planning), they will be represented at a level of detail at which that project was actually planned and controlled. An appropriate level of detail will be available for future planning, as the information requirements for a given type of work are defined in systems such as CACP.

4.4. Chapter Summary

This chapter described the concept and approach adopted for CACP and stressed the advantage that the adoption of standard information representation techniques provides in consideration of the data retrieval concepts of case-based reasoning. The next two chapters describe CACP in action, explaining the resulting functionality provided by the prototype. Chapter 5 presents the interface and program structure and implementation, while Chapter 6 discusses some example scenarios of the prototype that match with process followed in adding functionality throughout the prototype's development.

Chapter 5: Interface and Implementation

The prototype application developed to provide proof of concept for CACP has gone through several development stages as this research was conducted. The initial application, PRINEX (PRoject INformation EXplorer) (Froese and Rankin 1998), was developed as a prototype construction method management system. The PRINEX application was then used as the foundation for HiClass (Rackley, et al. 1998), a system that focused on the graphical interface requirements for integrated construction management systems. The principles defined in HiCLass were then adopted to the initial PRINEX system to form the core module of CACP, the first major development step. The CACP prototype was complete when the case-based reasoning module, the second major development step, was developed and integrated.

5.1. *Interface Requirements*

Having addressed planning functionality, the data structure required and the approach taken, we now look at the specific actions that a user will be executing which guides the development of the user interface and implementation. These actions are summarized as browse, edit and reason.

Browse:	view hierarchically structured project information, view details of project information, view relationships between objects, and view type libraries of project information.
Edit:	add, delete, or modify project information, copy and paste information from inside and outside current project, and establish relationships between objects.
Reason:	select the criteria for "case retrieval," and determine what will be added to the project information upon retrieval.

Two interface components are necessary to provide the required functionality of CACP. The first is a means of efficiently locating appropriate records from potentially large collections. Because of the aggregation and specialization relationships, a tree-view interface that organizes records according to these hierarchies, seems to be effective for this. The second required component is a means of entering, viewing, and modifying detailed information for each record. In addition to these components, the following list highlights some of the features also required of this user interface:

- Users can open as many explorers as they wish and can set each to view any of the system's kernel objects in a multi-document interface (MDI).
- The detailed browser side (right) can be collapsed or expanded, so that the explorer can be used for listing records, accessing details, or both simultaneously.
- The relationships between the kernel objects can be toggled as a portion of the explorer.
- Relationships between records can be set by selecting one record in the tree view and choosing related objects from a list on the detailed side.

- Sections of a tree view (nodes or branches) can be copied for replication in a separate explorer, convenient for building up specific project information from available general information or for copying similar information between projects.
- A tree view for type objects shows both aggregation and specialization hierarchies combined into one tree structure differentiated by corresponding icons (useful for type library browsing).
- Multiple queries are possible to display tree views of combined project objects structured by their relationships and filtered by their attributes.

5.2. *Interface Implementation*

Figure 5.1 is a screen capture of the resulting user interface demonstrating an "explorer" that provides both of the components (tree view and detail) within a single window. Working with extensive project information through a hierarchical tree view interface has the potential to become a major information interface for integrated project information systems. As stated in section 2.7, the interface provides the best 2-dimensional visual alternative available, while research continues investigating 3-dimensional alternatives including 3-dimensional CAD models (Rackley, et al. 1998).

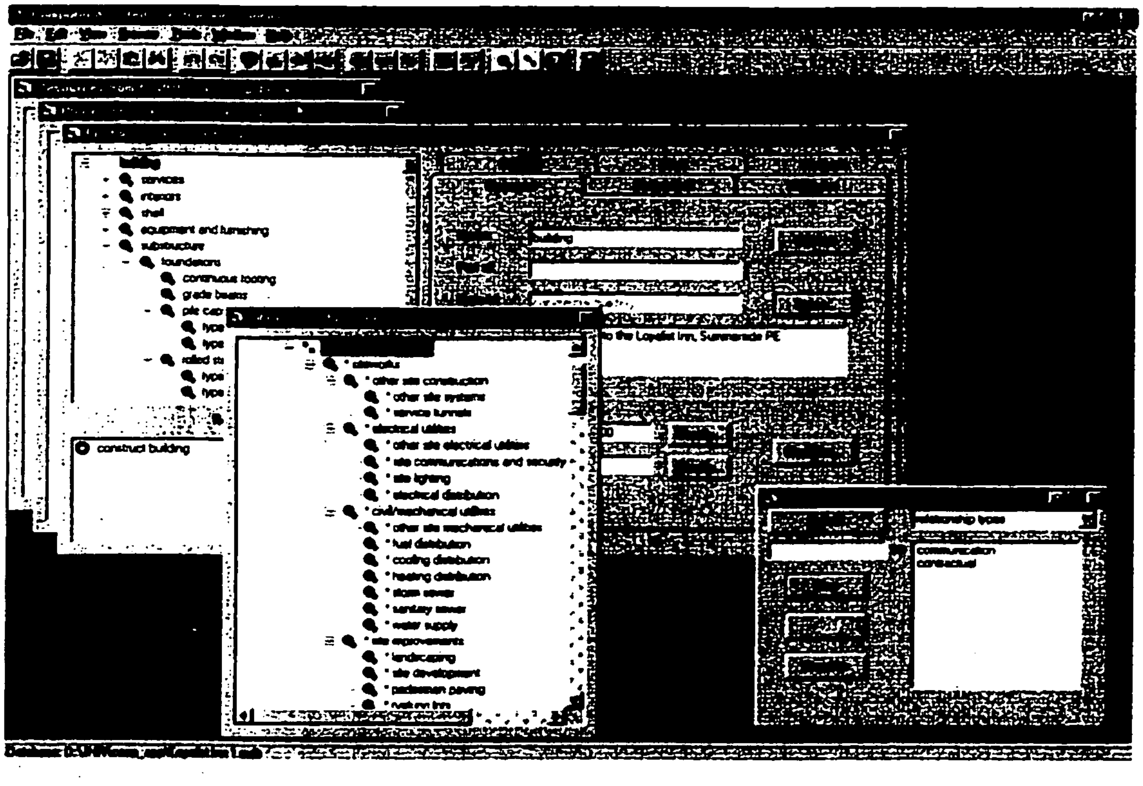


Figure 5.1: Overview of Computer-Assisted Construction Planning interface.

A variety of icons are used throughout the application to distinguish objects within the tree hierarchies and to indicate the functionality provided through a toolbar. Table 5.1 is a list of the icons found in the hierarchies (with the locations in which they are found and the concept they are representing), while Figure 5.2 gives a functional description for the toolbar buttons.

Table 5.1: Hierarchy icons, the source hierarchy and the concept they represent.

icon	name	Location	concept
	organization	agent hierarchy	Distinguishes organization agents from person agents
	person	agent hierarchy	Distinguishes person agents from organization agents
	cost	cost hierarchy	all costs are represented with this icon
	group	control hierarchy	indicates a group of controls
	assembly	control hierarchy	indicates an assembly of controls
	control	control hierarchy	indicates a single control
	resource	relationship treeview	indicates a process-resource relationship
	input	relationship treeview	indicates a product is an input of a process relationship
	output	relationship treeview	indicates a product is an output of a process relationship
	part	kernel object hierarchy	indicate a part relationship within a kernel object hierarchy
	type	library object hierarchy	indicate a part relationship within a kernel object hierarchy

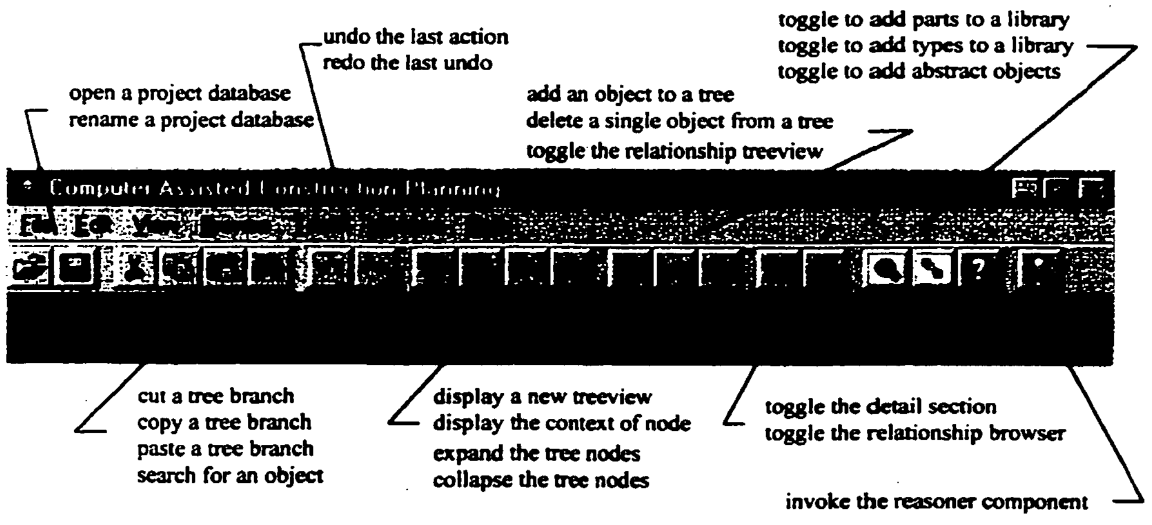
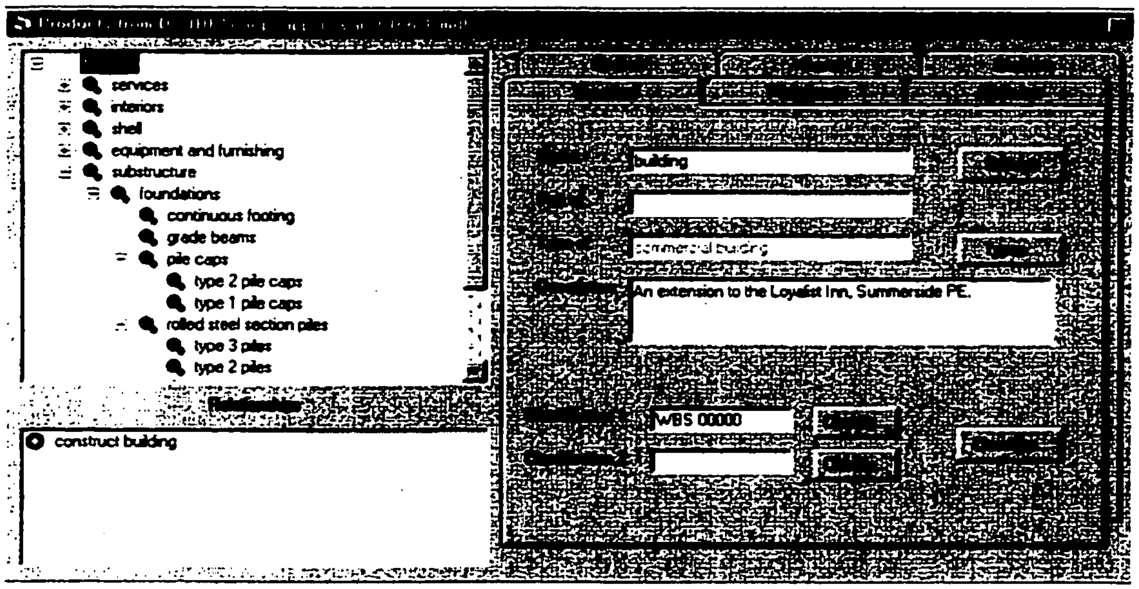

undo the last action
redo the last undo
toggle to add parts to a library
toggle to add types to a library
toggle to add abstract objects
open a project database
rename a project database
add an object to a tree
delete a single object from a tree
toggle the relationship treeview
Computer Assisted Construction Planning
cut a tree branch
copy a tree branch
paste a tree branch
search for an object
display a new treeview
display the context of node
expand the tree nodes
collapse the tree nodes
toggle the detail section
toggle the relationship browser
invoke the reasoner component

Figure 5.2: CACP's toolbar functions.

5.2.1 Browsing

Each "explorer" for a kernel object (products, processes, and resources) can be modified to display detailed information related to the selected node within the tree view. Figure 5.3 demonstrates a selected product within the tree hierarchy and the details for this selected product are organized on tabs (i.e., General, Participants, Attributes, etc.) in the right side of the "explorer" form. Details of the relationships between the kernel objects can be displayed in the bottom portion of the tree view side, also demonstrated in Figure 5.3 (e.g., the product *Building* is an *Output* of the process *Construct Building*).



building

commercial building

An extension to the Loyalist Inn, Summerside PE.

WBS 00000

Figure 5.3: Object explorer with details, *General* tab.

Buttons on the detail side of a kernel object "explorer" open additional "explorers" for the kernel object's related attributes. For example, Figure 5.4 is a screen capture of the "explorer" that is displayed when the *Control* button on a kernel object "explorer" is selected. These additional explorers have the same structure as those for the kernel objects (i.e., tree/detail). Specialized "tree/detail" forms are available for the objects *Classifications*, *Agents*, *Attribute Sets*, *Controls*, and *Costs*.

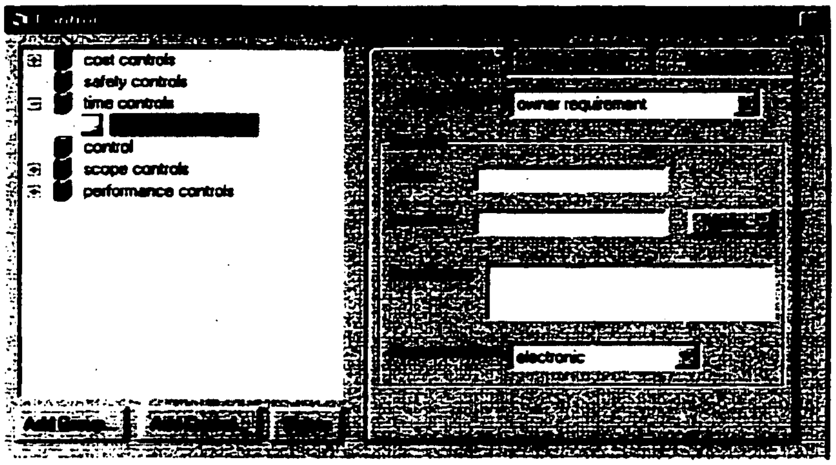
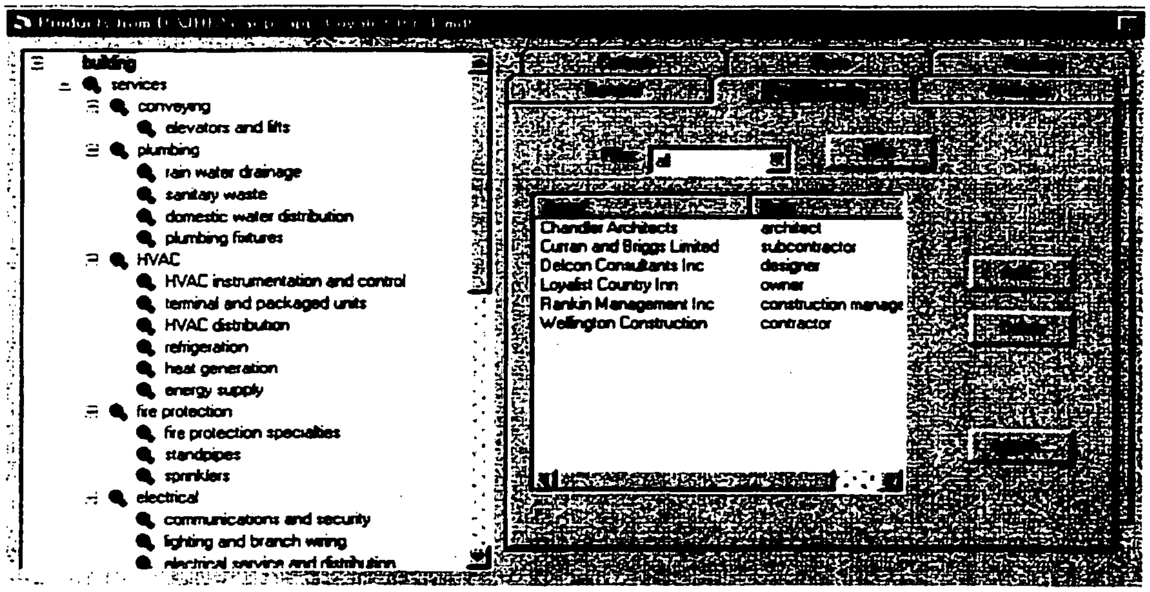


Figure 5.4: Explorer for Control objects.

5.2.2 Editing

Adding and deleting kernel objects in the simplest scenario is accomplished by modifying the appropriate hierarchy (adding or deleting nodes). Modifying a kernel object's details is also accomplished using the "explorer" and modifying an object's attributes. Modifying attributes to a kernel object is a matter of establishing relationships between kernel objects and other project information objects. Buttons are available to accomplish these tasks as shown in Figure 5.5.



Chandler Architects	architect
Curran and Briggs Limited	subcontractor
Delcon Consultants Inc	designer
Loyalist Country Inn	owner
Rankin Management Inc	construction manage
Wellington Construction	contractor

Figure 5.5: Object explorer with details, *Participants* tab.

Establishing relationships between kernel objects requires a different view of the kernel "explorer". In this case, a second tree view is used to represent the available objects and the *type of* relationship is defined using a pop-up menu as depicted in Figure 5.6.

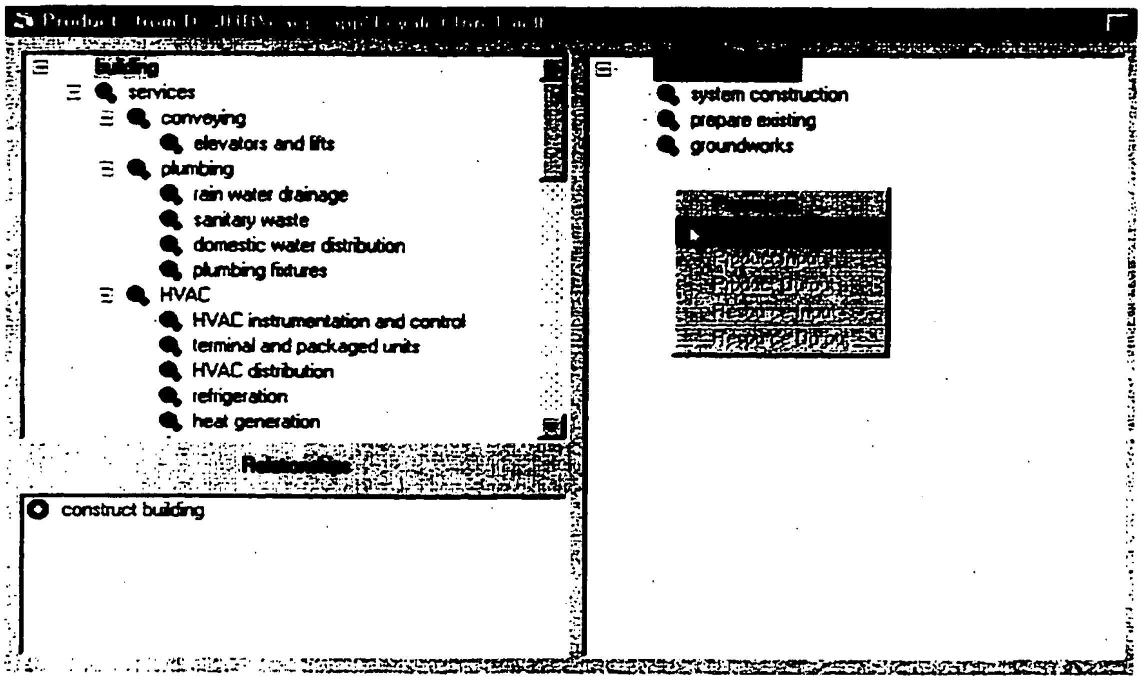


Figure 5.6: Object explorer with available relationships.

There are a variety of situations where implementing selection lists are useful to both guide the user and standardize the values to be selected. Although these values are limited to those available in the selection lists, the lists themselves can be modified. An interface is provided for this selection list maintenance as shown in Figure 5.7, which displays the current selection list of values for the relational object attribute *participant role type.*

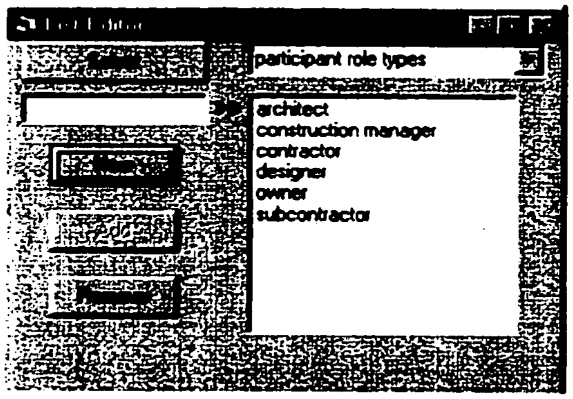


Figure 5.7: Selection list editor.

5.2.3 Reasoning

Reasoning is implemented to support the development of a plan by elaborating on products and considering methods for adoption to current project information. The *Planning Reasoner* form, as shown in Figure 5.8 guides reasoning sessions. There is a certain degree of flexibility available during retrieval where the user has the opportunity to determine attributes to be used for matching and their relative weighting.

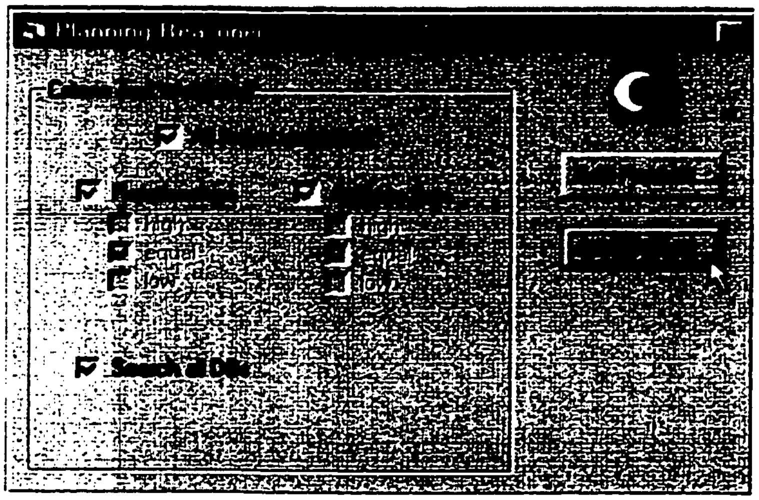
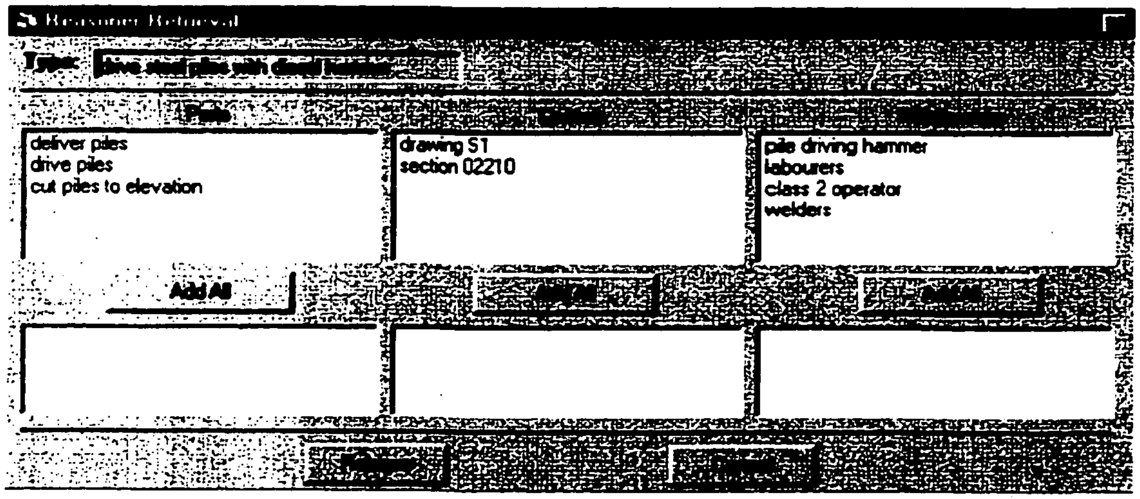


Figure 5.8: Defining criteria for case retrieval.

Once a case has been retrieved, the user will also need to decide the contents of the information that will be applied to the current project. This is accomplished through the *Reasoner Retrieval* form (Figure 5.9). Several other forms support the reasoning process and are described in Chapter 6.



deliver piles
drive piles
cut piles to elevation

drawing S1
section 02210

pile driving hammer
labourers
class 2 operator
welders

Figure 5.9: Determining information to be added.

5.3. *Implementation*

The CACP system is prototyped using Microsoft Visual Basic 6.0. By employing Microsoft Access the relational database engine is used and its tables are structured according to the PMDM. The case-based reasoner is The Haley Enterprises' "Easy Reasoner". The Easy Reasoner functionality is provided through a Visual Basic control and requires a minimal data conversion to flat dBase database files during a reasoning session (transparent to the user). The implementation tools were chosen for their ease of development (an important property of rapid application development tools) and integration characteristics.

As is the nature of Visual Basic, the data management "procedures" are separated from the interface code within the completed application. Code related to the reasoning features is also separated from the general data management code. A complete listing of the Visual Basic procedures and a form hierarchy is provided in Appendix B to demonstrate the scope of the prototype that has been implemented.

Figure 5.10 shows the three basic components of the implementation (Visual Basic, Easy Reasoner, and Access) and the following sections describe the contents of each component.

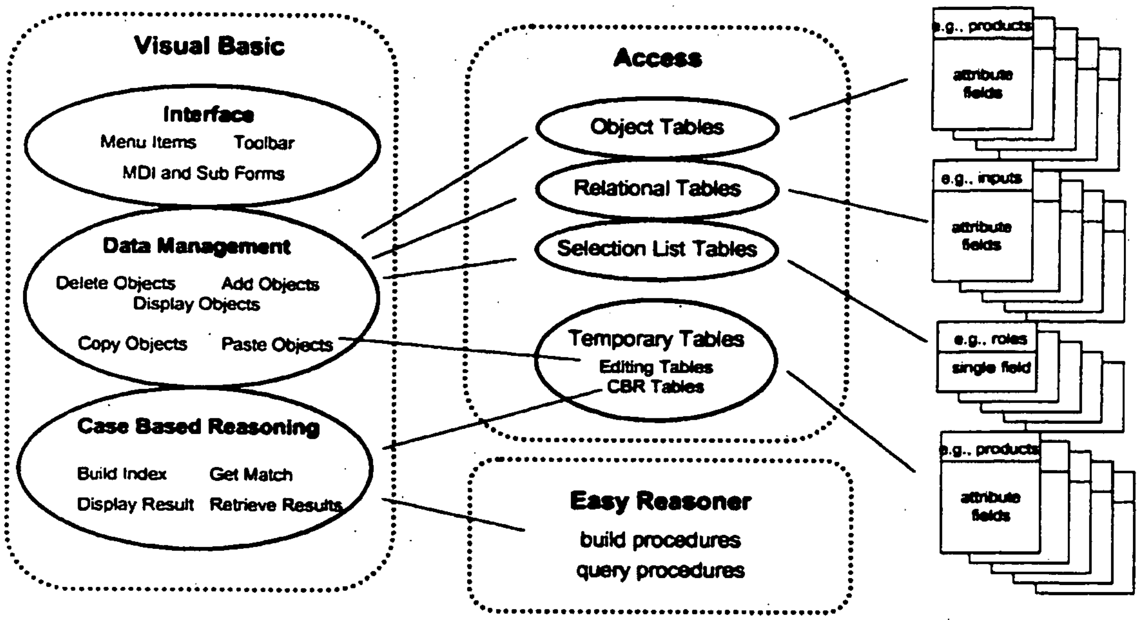


Figure 5.10: Software implementation scheme.

5.3.1 Relational Database

As stated previously, the relational database follows the structure of the PMDM. Implemented in Microsoft Access, each object from the PMDM essentially constitutes a relational table with its simple attributes and relational attributes. For example, the *Products* table contains simple attributes such as *description* and *name*, and also relationship attributes such as *part of* and *type of* that draw their values from designated table fields. Tables also represent objectified relationships with a unique identification; the key values from two tables, and any additional information required to describe the relationship, form the table fields. For example, the *Product Inputs* table consists of a unique value for *product input id*, and values from the *Products* table and the *Processes* table. Selection lists are also represented as tables, although they are simply single field tables.

5.3.2 Data Management

The code supporting data management comprises "procedures" addressing the basic browsing and editing actions: display, add, and delete. A procedure is available for each information object – action combination (e.g., *DisplayAgents*, *AddAgents*, *DeleteAgents*, etc.).

The *display* procedures essentially search the table in question and build a hierarchy representation based on the relationship attributes. While making a selection in the hierarchy (selecting a node) executes a procedure for retrieving and displaying the appropriate detailed information in the detail tabs.

The *add* procedures for objects determine the relationship attributes based on the position selected within a hierarchy and prompt the user for additional information through the use of dialogue forms. Once collected, the information is posted to the appropriate tables within the relational database. *Add* procedures for objectified relationships follow the same pattern although the relationship attributes are not relevant in these cases, the unique values are determined by the selected node in a hierarchy.

The *delete* procedures for single objects within a hierarchy run a check on related objects for potential deletion and upon confirmation remove the object and related object from the tree and database. Delete procedures are also available for tree segments, following a similar procedure as *copy* procedures described next and then removing the results.

Copy procedures are required to support the selection and reproduction of segments of kernel object branches. These procedures are recursive in nature, as they must navigate through a tree to collect the necessary information and store project information in temporary tables so the information is available for pasting. *Paste* procedures make use of the *add* procedures and draw information from the temporary tables.

5.3.3 Case Based Reasoning

Case-based reasoning functionality is implemented through procedures that support building cases, indexing cases, searching cases, and displaying results. The procedures *CBRBuildCases* draws upon information from all selected project databases and retrieves the relevant case information stored in relational tables together into a single flat table. This table is then transforms into a dBase file (the required format for the Easy Reasoner).

After transformation of the cases into a dBase file, the procedures make use of functions provided with the Haley tools for interacting with its inference engine. The cases are indexed using the procedure *CBRBuildIndex* that would take into account any weighting criteria for retrieval.

CBRGetMatch conducts the case based retrieval, performed according to nearest neighbour indexing otherwise known as a "query by similarity". The "query by similarity" ranks retrieved records according to their distance from the presented case. The distance is calculated employing the Easy Reasoner's technique of n-m grams for text retrieval and computes a nominal distance based on the weight assigned to each attribute and the values for n and m variables, as opposed to employing techniques such as fuzzy logic or statistical clustering algorithms. Computation of ordinal fields (integers, real number, dates) uses a similar calculation. Distance calculations are explained with an example in Appendix C.

After the distances have been calculated the user selects the case for possible retrieval and the procedure *CBRDisplayResults* is used to populate the *Reasoner Retrieval* form (Figure 5.9). The procedure *CBRAddMethod* adds the case information to be retrieved to the current project information in accordance with what the user has selected.

5.3.4 Interface

Procedures are also required to support the implementation of common windows-based software application features such as the menu bar, toolbar buttons, pop-up menus, and help features.

5.4. Chapter Summary

The prototype application developed addressed the interface requirements of the integrated planning tool. Development was accomplished using common software development tools with proof of concept in mind. The resulting interface proved sufficient in exploring the concepts behind CACP. The next chapter demonstrates these concepts through a planning example.

Chapter 6: Validation and Testing

Throughout the development of the research prototype sample project information was used to test the validity of the PMDM and interface. Information from an on-going construction project in which the author played a role in providing construction management services served first as a test of the application's browsing and editing capabilities as well as a validation on the structure of the database. The example project was an expansion of an existing hotel, the Loyalist Country Inn, in the city of Summerside, Prince Edward Island. The project entailed the new construction of a four-story building with a structural steel frame and wood in-fills, and a wood siding exterior. The space consisted of 52 rental rooms, extension of banquet facilities, a mechanical room, and an elevator. The building had an approximate budget of $2.2M and a construction schedule between June 1997 and June 1998.

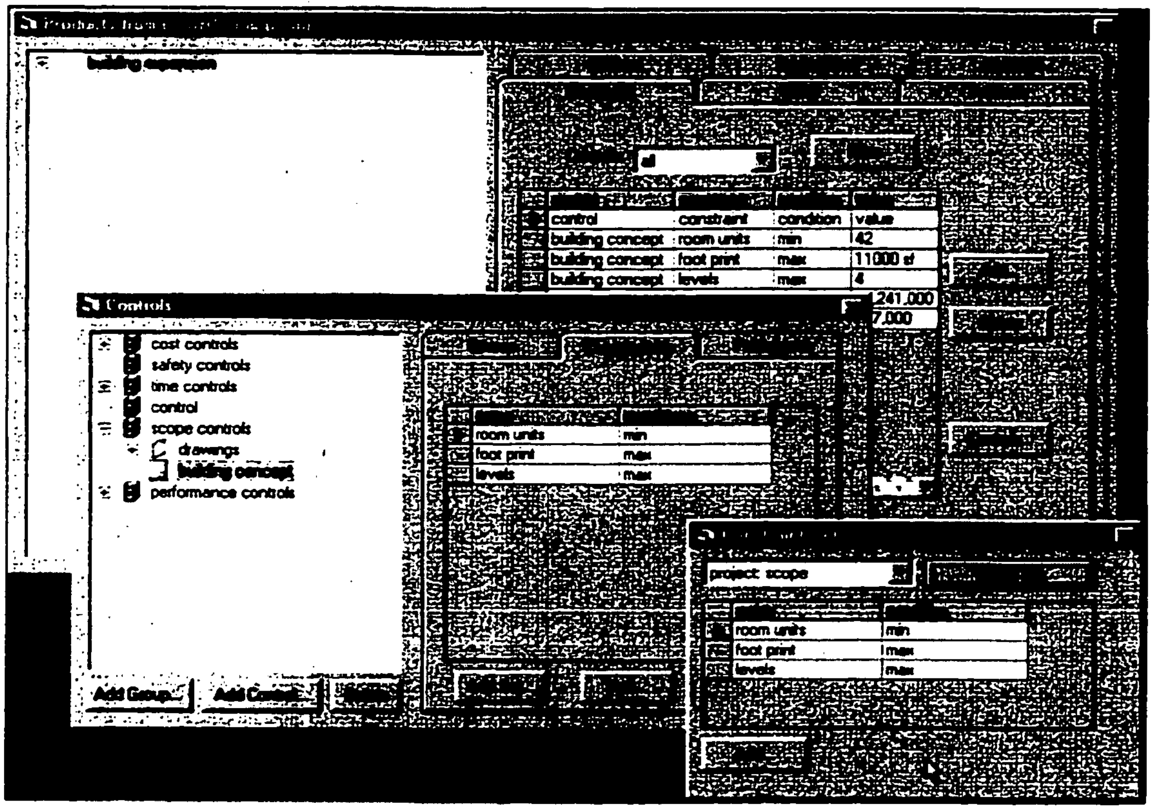
"Shells" of type information were then added based on the structure of industry sources, namely Uniformat (CSI 1998) for *product types* and Uniclass (Crawford et al. 1997) for *process types* and *resource types* to provide the structure of the kernel object libraries and guide the structure of project information. As much detail as possible was provided for at the highest hierarchical level of the project information, e.g., building expansion (product), construct building (process), and project resources (resource), to ensure the breadth of the model was adequate. For example, information related to each aspect of the kernel objects at this level was added (i.e., project agents, project attribute sets, project costs, project controls, etc.) and expanded where applicable (e.g., constraints sets where added for the project controls). The foundation system was then chosen as the area to expand on the existing high level of project information due to its relatively simple processes in comparison with other construction activities. This facilitated a narrow scope of information at a detailed level and served to demonstrate the two extremes in terms of the level of detail at which a user could work. Methods of foundation construction were the focus for an example of case representation and of how to support selection of appropriate construction methods as a means of further populating a project's information. As mentioned previously, the focus of the research was not to validate the contents of the case-base but rather to demonstrate the functionality provided by the reasoning add-on.

The scenarios below follow a progression in the description of how the prototype can be used to support the development of initial project planning information. The scenarios also correspond with the approach taken to add the required functionality during development of the prototype. The first scenario describes the steps involved in starting from scratch to build project information, while the second scenario addresses modification of existing information. The third scenario relies on the support of shells of information stored in the *type libraries*, while the forth scenario discusses the use of the case-based reasoning component to support the selection of methods and products from existing cases.

6.1. Scenario 1: Starting From Scratch

Starting with an empty database and populating it with project information was the first scenario tested as a means of validating the PMDM and the prototype's interface. The project information was developed beginning with the high level kernel objects "building expansion" (product), "construct building" (process), and "project resources" (resource). Detailed information was then added for each of these high level kernel objects, requiring the development of additional hierarchies of the kernels' attribute objects (agents, classifications, controls, etc.).

For example, Figure 6.1 is showing the *control* details for "construct building". In order to establish this link, the information for the appropriate *control* was added (in this case "building concept," a scope control), and the *constraint set* for this control also required development (in this case the "project scope"). Populating the project information for each of the kernel objects was then a matter of indicating the relationships between the kernel objects and their attribute objects. This is performed through each of the detailed tabs (i.e., Participants, Attributes, Controls, Costs).



control	constraint	condition	value
building concept	room units	min	42
building concept	foot print	max	11000 sf
building concept	levels	max	4

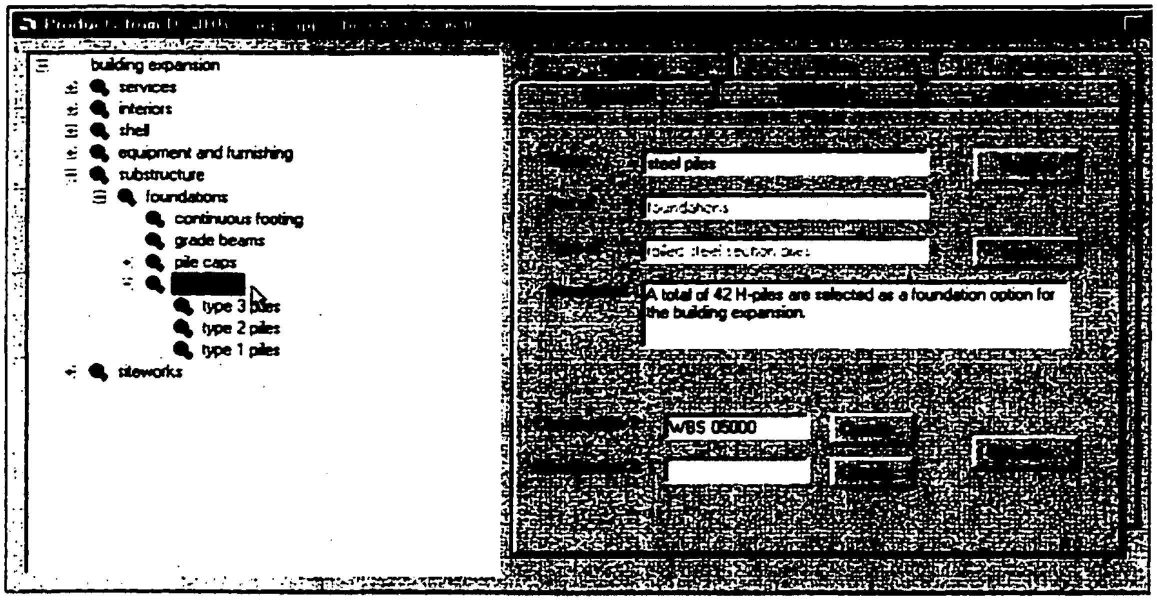
room units	min
foot print	max
levels	max

room units	min
foot print	max
levels	max

Figure 6.1: A high level product and path to detailed information.

In this scenario, the next step was to expand on the kernel *product* hierarchy by following one route of detail and expanding an aggregate hierarchy (i.e., all "part of" relationships). A similar pattern was also applied to the kernels' attribute objects. Figure 6.1 also depicts the development a *control* hierarchy that requires the development of its own attribute objects (e.g., *constraint sets*). As noted above, information related to the project's foundation system was the focus, and the hierarchy was developed down to the types of H-piles to be installed, shown in Figure 6.2. Completing these steps consisted of the development of an aggregation hierarchy of instantiated objects and their associated attribute objects.

Next the *process* and *resource* hierarchies were developed in the same way that the *products* and their attributes were. As the appropriate *processes* and *resources* were added to the project information the final step was to establish the relationships between the kernel objects. Figure 6.3 is reflecting the setting of relationships between the *process* "drive steel piles with diesel hammer" and the *resource* "pile driving hammer" as a *resource input* while the *product* "steel piles" has been previously related to the same *process* as a *product output.*



steel piles

foundations

[illegible]

A total of 42 H-piles are selected as a foundation option for the building expansion.

WBS 05000

Figure 6.2: The product hierarchy for H-piles.

6.1.1 Comments on Results

Building a project's information from scratch tested the usefulness of the PMDM in capturing a project's information and tested the prototype's interface in its ability to adequately support the input process and its representation of the information. The prototype performed both tasks of inputting and representation adequately. However, the representation of *controls* and their related attribute objects representing *constraints* did require some revisions both in terms of the PMDM and the interface.

Initially it was thought that *controls* could be treated the same as the other kernel objects both in their representation (PMDM) and use (prototype). It was revealed during testing of this first scenario that it is more appropriate to treat *controls* in a way similar to the other kernels' attribute objects. There are two reasons for this. The first is due to the fact that *controls* are represented as *records* and a relationship between *records* and *agents*, or *records* and *classifications* seems more appropriate and intuitive than relationships with *controls*, a more abstract concept. Secondly, *controls* are also broken down as *constraints* with these *constraints* being grouped into *constraint sets*. Relationships are still represented (established) between the *product*, *process*, and *resource* kernel objects and *control* objects in the PMDM, however it is the *constraints* and their values that are the focus from the perspective of representing this information to a user. Therefore modifications were made to treat *controls* as the other kernels' attribute objects in the implementation. However, this change only affects the interface, the data model remains unchanged from what is presented in chapter 3.

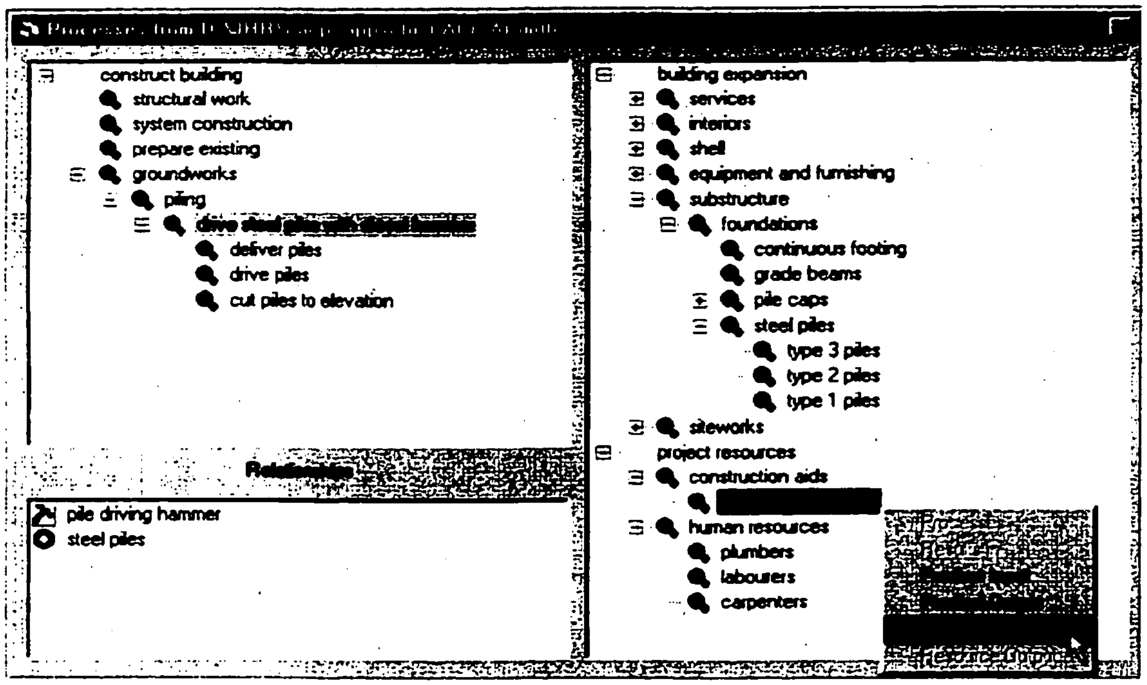


Figure 6.3: A process and its related kernel entities.

6.2. Scenario 2: Modifying a Current Project

The next step in development of a tool that would support the creation of an initial construction management plan was to enable the segments of an existing plan to be transferred to a new plan. As an initial step, this was to be performed without consideration of the relational attributes for objects within a given hierarchy. For example, Figure 6.4 is depicting two hierarchies of *product* objects from two different projects. Functionality was added to the tool to enable the user to select all, or segments, from an existing project hierarchy and to copy and paste those objects into a new project hierarchy. New project objects and their aggregation attributes are added to the database for the target project. However, the relational objects between kernel objects are not transferred through this function. For example, the *product* "foundations" and its complete *product* aggregate assembly in Figure 6.4 can be copied from one project to another, however its relational attributes (e.g., output of process) are not transferred.

6.2.1 Comments on Results

This feature was implemented and tested successfully and enables a user to quickly populate a new project's kernel object hierarchies. The functionality required development of some challenging code that is recursive in nature, as well as the addition of temporary tables to a project's database in order to capture the copied hierarchy segments. Capturing all the information that is rooted to one node in a tree structure requires a "depth-first" capture of information followed by a backwards progression to the root while ensuring that the entire depth is explored before a second backwards step. For example, consider capturing the information that follows *foundation* in Figure 6.4. As we step down to the level of *type 1 piles* a memory of where we have searched must be kept to ensure that all information at a given level is captured and that all paths are searched as we step back to the root (e.g., the subtypes of *pile caps* and potentially the sub-subtypes of *pile caps*). The temporary tables serve the purpose of capturing the information that tells us where we have been in the tree as the search progresses and at the end of the search the result is an index of the information to be captured.

The same code can be used on any of the hierarchies of project information but has only been implemented for the kernel objects. From a programming perspective the development of this functionality also provided a precursor to the library and reasoning functions that followed. This added functionality also required the implementation of pop-up menus to the user interface.

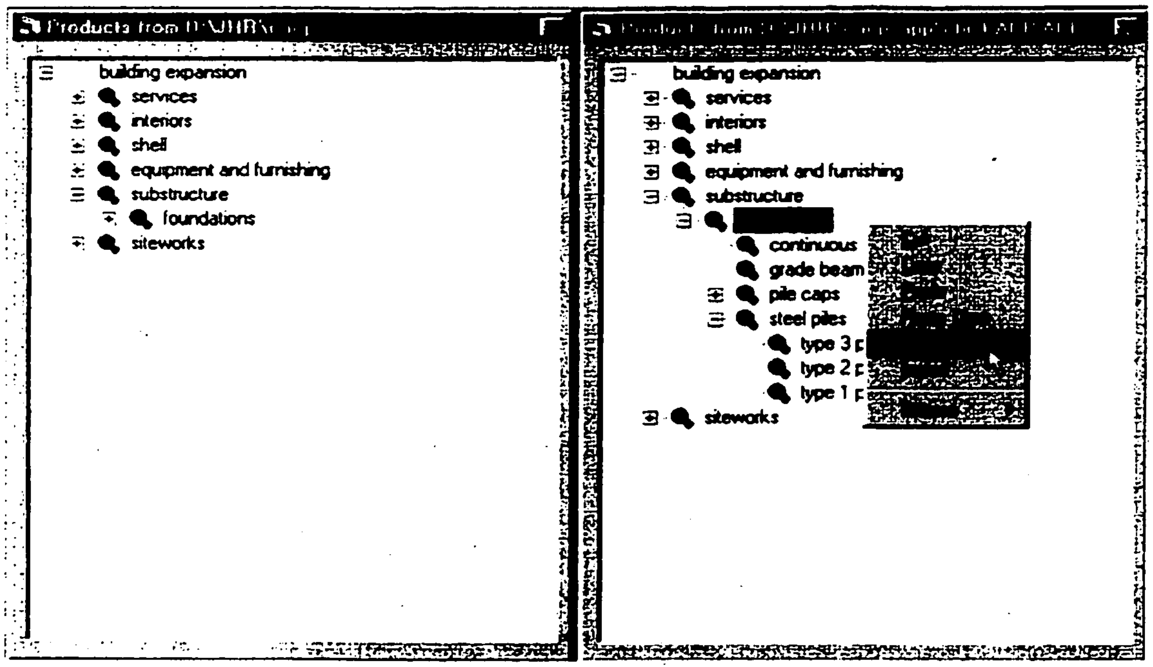


Figure 6.4: Copying from an existing project.

6.3. *Scenario 3: Building from Type Libraries*

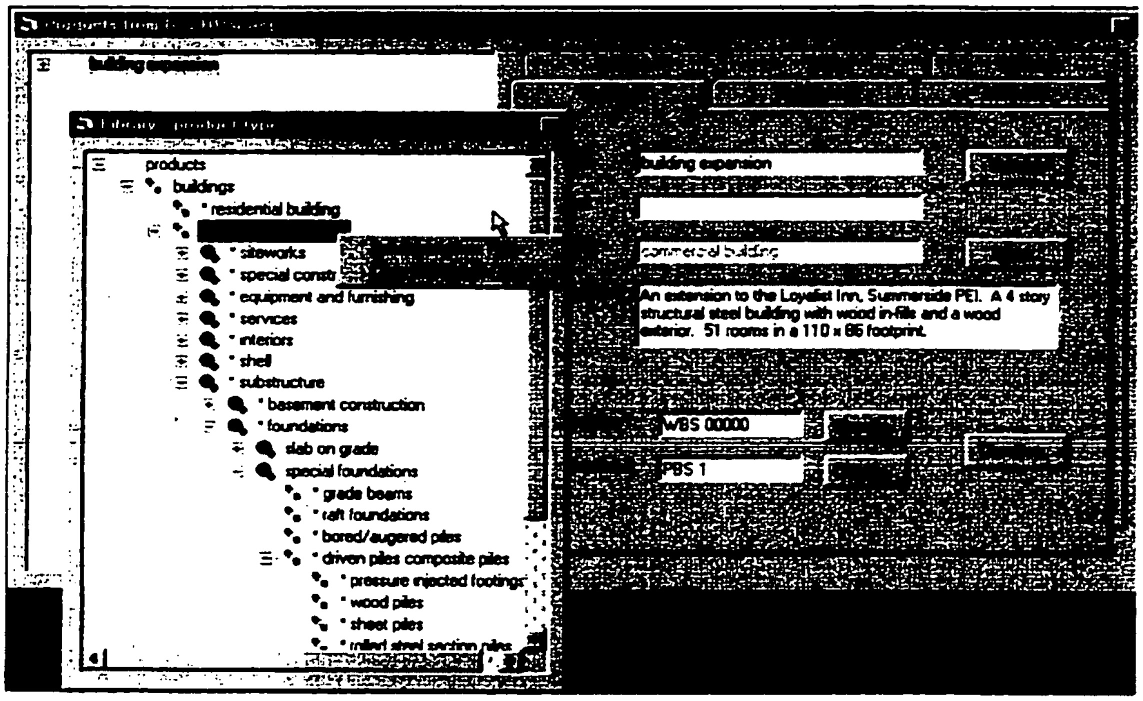
The third scenario for building up a project's information base is the use of libraries of stored *type* information. In this case, instead of referencing an existing project of instance information only (one possible selection), the user has the opportunity to examine a complete library of type information (alternatives for selection). Whereas the project information examined in scenario 2 was simply an aggregate hierarchy, *type libraries* are a combination specialization/aggregation hierarchy containing both "type of" and "part of" relationships. For example, in Figure 6.5 the product library is displayed showing a specialization relationship between the *product* "buildings" and the types of buildings "residential building" and "commercial building," while an aggregation relationship is displayed between "commercial building" and its parts "siteworks," "special construction," etc. As indicated in Table 5.1, there are two icons used in the library hierarchies to denote the "part of" and "type of" relationships between objects. The *type libraries* also contain abstract objects that are essential to organizing a library of information, such as the product "special foundation". These abstract objects are used to give the library structure but are not expected to be instantiated in a project.

The first of two ways to use an object type is to set an existing project object as a type. For example, the user could set the *product* "building expansion" as a type of "commercial building". Alternatively, the user could use the library as a source of populating the current project by copying a non-abstract object from the library and pasting an instance of this type object into the project. For example, if "siteworks" were to be included in the building expansion then the *product type* "siteworks" could be instantiated into the new project.

By building a project through the use of type libraries the user has the ability to also populate the new kernel objects with the attribute objects such as *attributes, classifications*, and *costs*. This is a progression from the first scenario where only the kernel objects were added to the target project, however it still does not address the relational attributes, the focus of scenario 4.

6.3.1 Comments on Results

As stated previously, the *type libraries* were structured using several industry sources of classifying information (Uniformat (CSI 1998) for *product types* and Uniclass (Crawford et al. 1997) for *process types* and *resource types*). The majority of the effort was spent on developing an effective combination *aggregation/specialization* hierarchy for each kernel object rather than fully developing the supporting attribute detail for each type object. The combination hierarchy proved very useful in this role, allowing a user to efficiently search through very large structures of type information for possible selection. Also, in the current implementation the type information is stored in each database of project information, however this would not be the case with the full development of supporting attribute details. In full development, each *type library* would be expected to have its own database.



building expansion

commercial building

An extension to the Loyalist Inn, Summerside PEI. A 4 story structural steel building with wood in-fill and a wood exterior. 51 rooms in a 110 x 86 footprint.

WBS 00000

PBS 1

Figure 6.5: Selecting from a library of type information.

6.4. *Scenario 4: Reasoning About Methods*

Scenario 4 implements case-based reasoning and also addresses the selection and addition of relational objects to the target project. This scenario considers a *product* in the target project and supports the user in selecting an associated method (process, controls, types, etc.) by deciding what comes along with a selected *process type* into the current project.

As an example, consider the *product* "steel piles" that is part of the foundation for our current project and the selection of the *process type* "drive steel piles with diesel hammer" from a library of process types. At this point the scenario is no different than that presented in scenario 3, however using the functionality provided by case-based reasoning it is possible to add information related to the process type to the current project. To accomplish this a "reasoning" session is invoked. Figure 6.6 depicts the set-up form for a reasoning session. Before executing a retrieval of cases, it is possible to weight the attributes upon which the distances between the presented case and each case retrieved will be based. Again, because of an under-developed case-base, the current weighting is equivalent for all attributes.

A query is performed based on the case information presented and the criteria and weighting values, and each case considered is ranked according to its distance from the presented case. For example, "reasoning" about adding a method for "steel piles" presents a case with *process type* "drive steel piles with diesel hammer," *product type* "steel H piles," and the various attributes and their values associated with the product (e.g., length, section area, etc.) (Figure 6.7). This information forms the presented case and is used to calculate the distances to all cases retrieved from the case base.

To build the cases for comparison a search is made of existing project databases for the presented *process type* and *product type* and the relevant information (attributes and values) is retrieved for each of the potential solutions along with an index back to the original source. The result of the distance calculation of each case to the presented case is then displayed to the user (Figure 6.8).

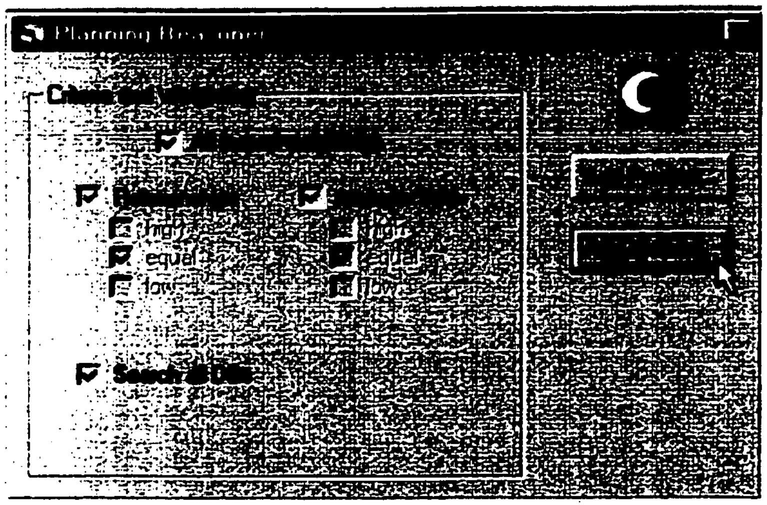
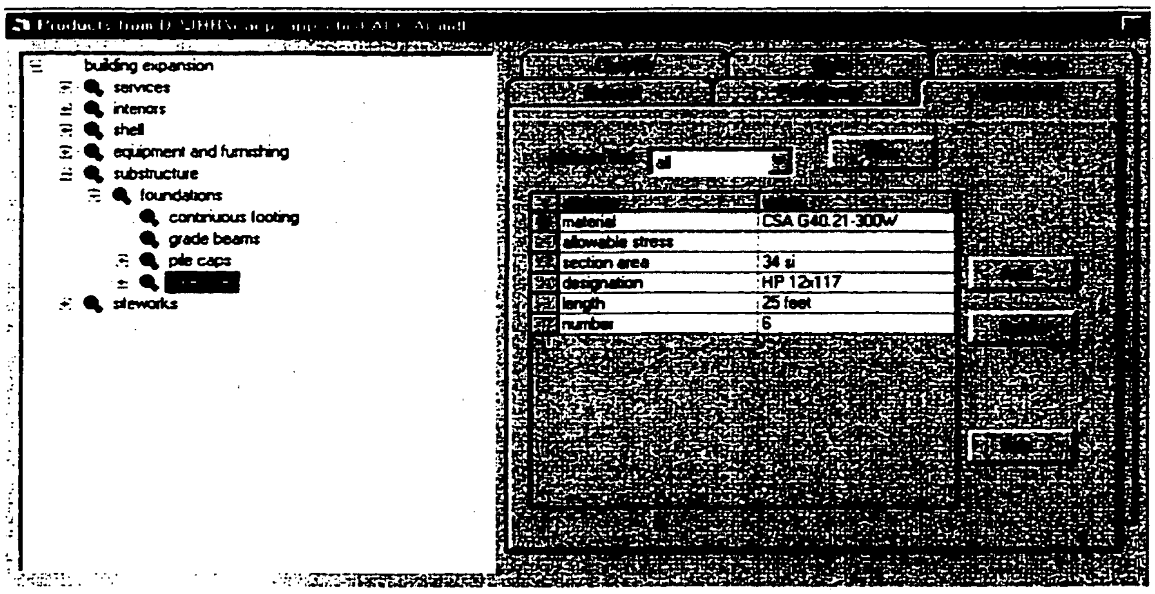


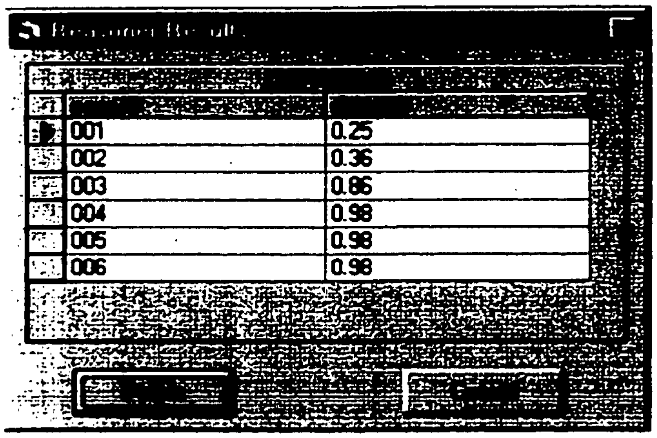
Figure 6.6: Set-up for reasoning about methods to select.



- building expansion
  - services
  - interiors
  - shell
  - equipment and furnishing
  - substructure
    - foundations
      - continuous footing
      - grade beams
    - pile caps
  - siteworks

material	CSA G40.21-300W
allowable stress	
section area	34 si
designation	HP 12x117
length	25 feet
number	6

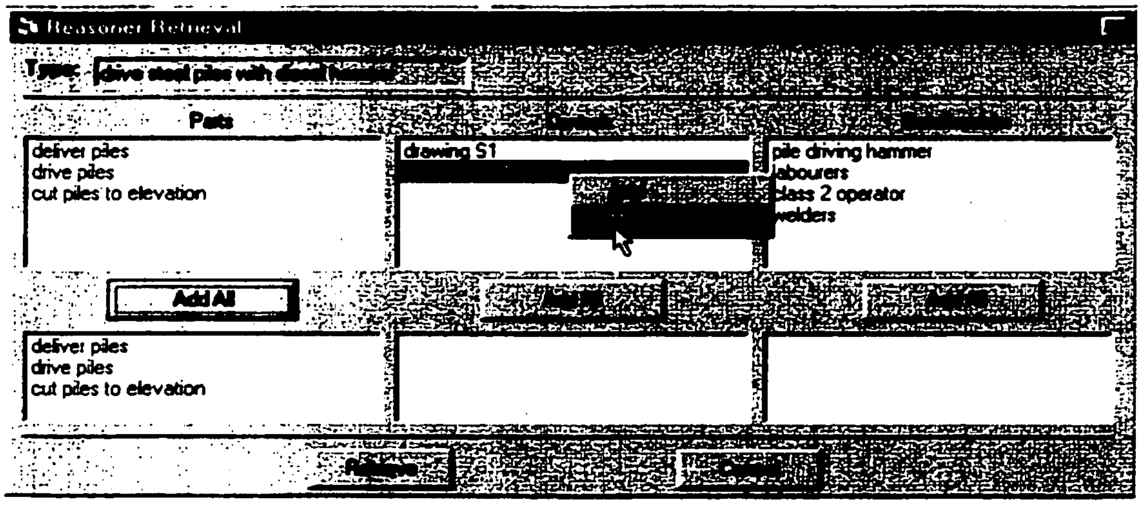
Figure 6.7: The reasoning product and its attributes.



001	0.25
002	0.36
003	0.86
004	0.98
005	0.98
006	0.98

Figure 6.8: Results of query by similarity.

The resulting case that returns the closest distance to the case presented is selected and results in the retrieval of a case (see Appendix C for distance calculations). For example Figure 6.9 shows the results of the method returned for the *product* "steel piles". The method consists of the *process type* "drive steel piles with diesel hammer" and the associated *process* sub-parts, *controls*, and other relational objects to be considered for retrieval and thus addition to the current project.



Reasoner Retrieval

Type: drive steel piles with diesel hammer

Parts

deliver piles
drive piles
cut piles to elevation

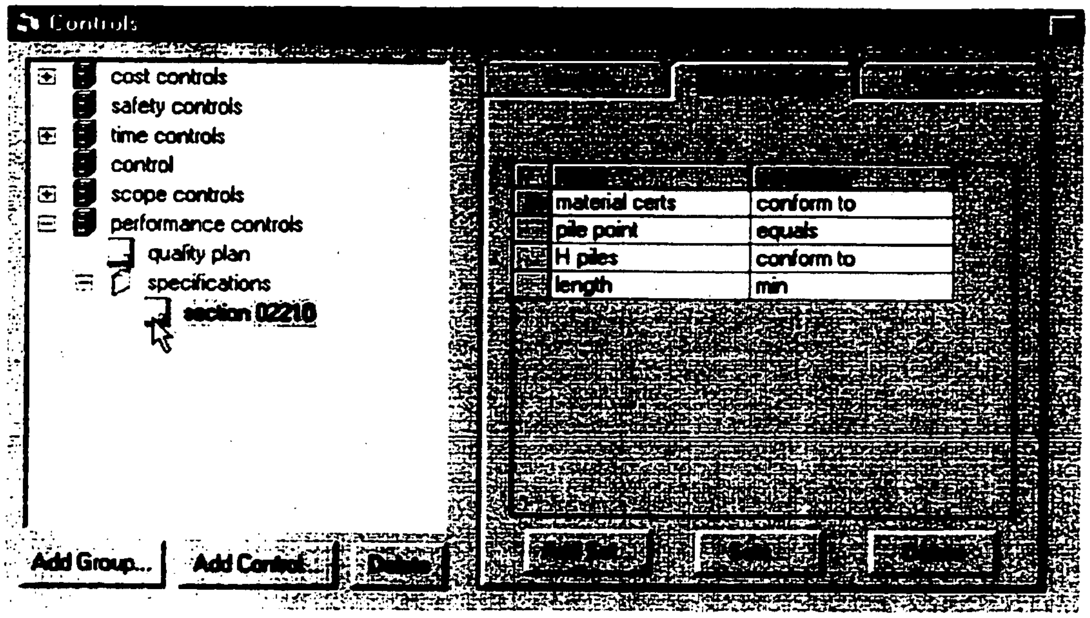
drawing S1

pile driving hammer
labourers
class 2 operator
welders

deliver piles
drive piles
cut piles to elevation

Figure 6.9: Deciding what information to retrieve.

While deciding what relational objects to bring into the current project from the retrieved case the user also has the opportunity to examine the detailed information relating to each potential object. For example, before selecting the *control* "section 02210," shown in Figure 6.9, the user may wish to examine the details behind this control by opening an additional explorer (Figure 6.10) with the case project as its source.



material certs	conform to
pile point	equals
H piles	conform to
length	min

Figure 6.10: Examining more detail prior to retrieval.

All selected information from the "Reasoner Retrieval" form is then added to the current project information, including the relationships between the kernel *product* object and the attributes selected.

6.4.1 Comments on Results

The number of cases used for retrieval was not sufficient to adequately examine the effectiveness of manipulating the weighting of relational objects versus attribute sets in the retrieval process. Therefore, all relational objects for the kernel object considered and all attribute values are considered during the retrieval process. Also, rather than focus on the validity of the information stored in the cases used, the prototype concentrated on development of the functionality required to complete the steps presented.

6.4.2 Reasoning About Products

Elaboration on a product answers the question: what type of product will be used? The answer is based on a comparison of the existing project controls and attributes pertaining to a higher level *product* and leads to population of the current project information with sub-parts, complete with their relational objects. In the scenario presented for CACP this question is probably already answered, as the assumption is that we already have a product breakdown. This is parallel with the support that could be offered to the development of a design, and the case-based reasoning solutions discussed in section 2.6.1.

6.5. Chapter Summary

The scenarios presented in this chapter demonstrate a progression in the support of the planning process that can be offered by a system such as CACP, and also gave a glimpse into the addition of functionality throughout the prototype development. The next chapter highlights the contributions made by this research. The system provides a method to build up the initial information requirements for construction planning in the context of an integrated construction management system. The result is the population of an integrated project database with information that can then be used as a starting point for more detailed planning tools.

Chapter 7: Contributions and Conclusions

Overall, the research recognizes the future requirements of integrated construction management systems, as they have been foreshadowed in recent research, and addresses the need to support the management of large volumes of information on several levels. The solution proposes a combination of providing an efficient user interface while exploring methods to support the partial automation of the required information through the access of stored information for past projects.

The completion of this research has resulted in the development of a prototype construction management system with the primary characteristic of assisting the user in the manipulation of information in order to generate the initial information requirements of an integrated construction management system. The research has employed sound information representation principles, explored the implementation of data management tools and techniques, and followed a rigorous development process.

The research recognizes and follows the path being established for integrated construction management systems that rely on a standard representation of the industry's information requirements. The core information structures were adopted and their validity confirmed by examining a broad application for an integrated system (i.e., planning information requirements beyond time and cost). By examining the comprehensive aspects of construction planning in light of the framework for an integrated construction management system the research demonstrated the usefulness of applying sound information representation structures (i.e., part and type relationships). Through the application of prudent information systems development approaches (i.e., data analyses , process analyses, and interface analyses), and the use of case-based reasoning, the research has advanced the concepts of planning tools as they apply to integrated systems.

7.1. *Overall Contribution*

In terms of new contributions over previous research efforts, the functionality provided by the developed tool must be viewed in the context of the conceptual definition of integrated construction management systems. The foundation for integrated construction management systems is currently being laid and the elements of CACP are targeted to directly support this effort. The additions to the data model and its representation of a broad scope of information required for the project management planning processes as applied to construction, along with the implementation of this data model, are unique contributions. The representation of construction management information in hierarchies, the partitioning of information from the perspectives of products, processes, and resources, and the practice of capturing and reapplying planning knowledge from past projects are not new concepts. However, these concepts have been combined in a new class of construction management application that provides a comprehensive description of project planning information for initially populating and effectively managing a project database. The combined results are unique in that they deliver the required functionality of a general browsing and information management application that interacts with comprehensive project planning information. The tool's "tree-detail" interface, which is common to all central project objects, facilitates the population of the project information where building-up the hierarchies of project information can be considered *planning*, although not to the extent that additional components (e.g., scheduling applications, estimating applications, etc.) of an integrated system are expected to offer.

With the foundation created by the tool, opportunities are now present for further development of integrated construction management tools that focus on solutions addressing the complete characteristics of "Total Project Systems". There now exists a data model and an example of its implementation upon which the architectural characteristics required of integrated systems can be explored and tested. Aspects of an integrated construction management system addressing issues such as a multi-user environment and a seamless interaction among tools are two examples of these architectural issues. The completed research also provides the basis for examining and rethinking the application of more advanced tools and their planning processes such as the ramifications of introducing the functionality of using the combined hierarchies of information objects as a primary interface for planning.

7.2. Integrated Specific Contributions

Upon completion of this research, each of its stated objectives was addressed. Addressing each objective resulted in various contributions throughout the research. The significance of these individual contributions is summarized below and reinforced, where applicable, with explanations of the impact that this research has already had on similar work.

7.2.1 Contribution to Model Development

Objective: To contribute to the development of a standard information model for the construction and related industries.

In light of an integrated system approach, data model development took a broad view of project/construction management apart from the traditionally supported views of cost and time. Software applications supporting estimating, cost control, and scheduling have a solid presence in the industry while applications supporting complete views of organization, performance, documentation, and other aspects of construction management are emerging. For example, systems that focus on the performance of the construction process require information detailing the constraints imposed by controls on a project as well as an explicit representation of project participants their specific responsibilities and roles. The PMDM takes account of all these existing and potential supporting applications by concentrating on the functionality required of the construction management "planning" processes.

With this broad view, the PMDM served as a useful test in determining the required scope of a project management core model. It has attempted to addresses the key requirements of all project management application domains through the support offered by the kernel objects (product, process, resource, control), classifications, constraints, attribute sets, participants, and records. However, the depth of a project management data model can only be developed and verified by considering individual application domains (e.g., estimating, scheduling, etc.), again in the context of an integrated system, and beyond the scope of this research. It has determined the information requirements of an application that supports planning systems in integrated systems.

The most significant proposed modifications to existing industry standard data models is in the area of modeling controls, constraints, and records, and their relationships to the kernel and agent objects. To date, modeling efforts have not taken the step to break down a standard industry document into controls that can constrain specific aspects of a project. The most likely reason that it has yet to be supported in a model is that this technique has not yet been practically applied. In such a scenario, each project document would have to be explicitly linked to each applicable product, process, and resource in the project. For example, a concrete specification section or possibly subsection would have to be broken down into constraints and each constraint would have to be examined against each kernel object related to concrete to determine its applicability. This is a daunting task, considering the potential scope of a project specification and the other controls that make up a project, however solutions are being sought.

The PMDM was initially developed as a draft data model for the TOPS project and was useful in exploring many of the concepts related to the representation of the construction management domain. Work that was originally collaborative in nature has been extended for this research to address the specific issues surrounding an integrated approach to construction planning and examine breadth required of the model. The work complete for this thesis in the development of the PMDM also contributed to the aforementioned Facility Management Turnover System (Froese et al. 1997a). It also directly provided the foundation for an on-going NRC IRAP research project for the development and implementation of an Internet based collaborative project environment (CPE) where construction project participants can manage and exchange data through an on-line database (CTCA 1998).

Research continues in specific areas dealing with the depth of the model within the construction management domain by work projects within the IAI and research performed by Froese, Yu, and Gorlick (Yu and Froese 1999, Gorlick and Froese 1999) at the University of British Columbia.

7.2.2 A Supporting Component for an Integrated System

Objectives: To develop an effective/efficient interface for dealing with large quantities of information. To develop a hierarchical framework that facilitates the representation of planning information from separate views at different levels of abstraction. To implement selected components of a system that can generate an initial construction plan for use in an integrated construction management system.

The data model was implemented to investigate the application of planning tools in the context of an integrated construction management system. Prototype development focused on a front-end application that supported more detailed construction planning by providing the functionality required to initially populate a project database with planning information.

The interface developed for CACP has demonstrated how a 2-dimensional combination tree-detail view can be used for the representation of all types of construction management information and the management of this information. This type of interface should be intuitive to the construction industry whose members are familiar with the hierarchical representation of project information.

Although the modifications to this particular 2-dimensional interface (partitioning by the kernel and a tree-detail combination) have addressed the focus-context problem in representing and navigating large hierarchical structures of information, its practical application remains untested. Interfaces employing 3-dimensional approaches and 3-dimensional CAD show promise for more effectively representing the same information and also providing an added benefit of visualization. However, the interface developed proved more than adequate for this research and, as a result of the groundwork performed during CACP's development, research continues in this area by Waugh (1999), in an effort to produce a more effective and intuitive interface for integrated construction management information systems.

The same 2-dimensional interface has also been successfully adopted for use in the CPE project described above and, most recently, for an integrated prototype Quality Management Tool developed for the Construction Association of Nova Scotia and its membership.

7.2.3 Managing Existing Planning Information

Objectives: To explore an approach to managing planning information by applying case-based reasoning to construction planning.

The research has explored the use of case-based reasoning to computer-assisted planning and determined that it holds much promise as a method of simplifying the maintenance of so-called knowledge-based planners. In general, case-based reasoning methods are advantageous where a formal set of rules or algorithms for generating solutions is difficult to obtain but examples of correct solutions are readily available and where the flow of control over the retrieval system is not predictable. Case-based reasoning uses indexing as a method of eliminating unnecessary computations and relies on a well-engineered case-base to avoid duplicate cases from being retrieved.

Previous research has proposed approaches to alleviate what has been referred to as the "elicitation bottleneck" by developing well formulated models for representation. While many have described how this may be achieved, technologies and methods vary so much that it is difficult to capture all the possibilities that the special conditions surrounding each new project will create. Therefore, the maintenance of highly structured decision trees or IF-THEN statements defining the flow of control through a program is a challenge. The case-based reasoning approach simplifies the elicitation process by looking beyond the need or maintenance of hard coded rules for the representation of industry knowledge.

In a scenario where project information is stored in a format consistent with a common information model, it is possible to acquire new cases from existing data and acquire these cases incrementally. This is very beneficial in the addition of new examples (cases) of project information and in the retrieval of cases based on partial project information.

CACP's identification of solutions is by retrieval only (adaptation is not automated). This approach requires trained or expert planners who can determine if the retrieved information is relevant. Therefore, the use of case-based reasoning alone is not the sole answer as a limited number of rules could support the retrieval process by guiding the user through a search and thus simplify this process, as well as providing support to the steps required for the adaptation of retrieved cases.

The research has applied the concept of using partially generated plans or templates of planning knowledge to give the construction planner a head start in the planning process without hiding any of the knowledge that is required for good planning. Again, this is a combination of sound information modeling concepts coupled with beneficial interface functionality supporting the manipulation of hierarchical information as well as the information handling requirements of an integrated system.

This research is another useful example supporting the trend set by researchers such as Aalami and Fischer (1996), Chevallier and Russell (1998), and Jagbeck (1998) of a system that provides the "planner" an effective tool without attempting to entirely automate the planning process. This is important, as completing this "planning process" is an essential step in the successful management and therefore delivery of a construction product. CACP employs this concept in the context of an integrated construction management system, whereas several software tools available in industry also support the concept, but from a time planning perspective only. The trend is in recognition of the way people plan, starting with segments of past plans and building a complete plan to fit the current situation.

7.3. Conclusions

It is encouraging to know that research is on going world wide in many of the areas addressed by this thesis. Integrated construction management tools have been accepted as paramount to improving the productivity and efficiency of the construction industry, the largest industrial sector in most developed countries. The following recommendations are offered in light of the research completed, with the optimism that they will be useful for current and future related research.

7.3.1 Implementation and Field Testing

More work is required to field test the prototype. The best method to assure that research is addressing the needs of the industry in the case of construction management tools is to conduct field-testing of prototype applications. Successful field-testing usually consists of implementation partnerships with companies who see the benefit of exposing themselves to the latest advances of construction management technology. In light of the case-based reasoning approach, the testing should also focus on developing a case base in a specific area of construction.

7.3.2 Data Model

During this research the efforts towards developing standard information models for the purpose of easing integration across software applications within the industry has most definitely grown. Particularly interesting is the growth in support of construction and project management domains for applications such as cost estimating and scheduling. If an environment of integration is to be successful then these efforts must be challenged by a variety of industry participants, so that they can be solidified and strengthened. This requires contribution from all researchers in the domain of construction management, as well as software application developers and ultimately practitioners.

7.3.3 Interface

The 2-dimensional tree hierarchy is currently an adequate interface for an integrated construction management system. However, exploration of 3-dimensional solutions should also be pursued and early work points to several obvious advantages such as a more effective solution to the focus+context problem and an ability to visualize the construction product and its process. One challenge is that any solution must be accompanied with a practical implementation and even the 2-dimensional approach has yet to become an intuitive solution throughout the construction industry.

There is also a need for a distributed application solution and accompanying interface. Complete integrated construction management systems would provide detailed functionality beyond initial plan development (the focus of CACP) and the resulting interface should appear seamless to the user as they embark on using these supporting tools.

7.3.4 Reasoning

Future work for the reasoning functionality requires the results of field-testing and actual implementations to capture project information in an integrated database and thus serve to populate the case base.

It is also expected that future implementations will require the addition of rules to improve the reasoning process, and increase efficiency of search. A limited number of rules for constraint satisfaction could guide selection from a system perspective and improve the user's understanding of why cases have been retrieved.

Augmenting the case-based reasoning retrieval process with rules to address higher-level questions regarding context and domain, can also avoid problems of duplicate cases, or overlapping subsets of information cases. Additions can be made to the reasoning process (e.g., decision tree and hierarchical retrieval) to address this. However, all these suggestions will only be possible after additional testing.

References

1. Aalami, F. and Fischer, M. (1996). "Requirements for Industry Applicability of Knowledge-Based Schedulers." ASCE Proceedings of Computing in Civil Engineering, Anaheim, CA.

2. Ahmad, I., Russell J., and Abou-Zeid, A. (1995). "Information Technology (IT) and Integration in the Construction Industry." Construction Management and Economics, Vol. 13, No. 3, pp. 163-171.

3. Augenbroe, G. (1995). "The Combine Project: A Global Assessment." Computing in Civil Engineering: Proceedings of 2nd International Computing Congress ASCE, pp.163-171.

4. Autodesk (1995). Industry Foundation Classes: Management Overview. Industry Alliance for Interoperability, Autodesk, San Rafael, Calif.

5. Bilgic, T., and Fox, M. (1996). "Case-Based Retrieval of Engineering Design Cases: Context as Constraints." Artificial Intelligence in Design '96, Kluwer Academic Publishers, pp. 269-288.

6. Bjork, B.-C., (1992). "A Unified Approach for Modeling Construction Information." Building and Environment, Vol. 27, No. 2, pp. 173-194.

7. Breuer, J. and Fischer, M. (1994). "Managerial Aspects of Information-Technology Strategies for A/E/C Firms." ASCE Management in Engineering, Vol. 10, No. 4, July/August, pp. 52-59.

8. Brown, A. (1996). "Construction Modeling Methodologies for Intelligent Information Integration." World Wide Web document at http://www.salford.ac.uk/iti/projects/ commit/commit.html

9. Cassidy, M., (1999). "An Integrated Information Database for the Construction Industry." The Construction Specifier, CSI, Vol. 52, No. 7, pp. 43-48.

10. Cherneff, J., Logcher, R., and Sriram, D. (1991). "Integrating CAD with Construction Schedule Generation." ASCE Computing in Civil Engineering, Vol. 5, No. 1, pp. 64-84.

11. Chevallier, N. and Russell, A. (1998) "Automated Schedule Generation." Canadian Journal of Civil Engineering, Vol. 25, No. 6, pp.1059-1077.

12. Chin, S., Stumpf, A., and Liang, Y. (1996). "Object-Oriented Construction Information Framework for Construction Management." ASCE Proceedings Computing in Civil Engineering, Anaheim, CA, pp.786-792.

13. Computer Associates (1997). Platinum Process Continuum, product information at http://www.lbms.com/products/appdev/ppcpr.htm, first visited September 1997, last visited December 1999.

14. Construction Specifications Institute (CIS) (1995). Masterformat – Master list of Numbers and Titles for the Construction Industry. Alexandria, VA.

15. Construction Specifications Institute (CIS) (1998). Uniformat – A Uniform Classification of Construction Systems and Assemblies. Alexandria, VA.

16. Construction Technology Centre Atlantic (CTCA) (1998). "Secure Document Handling System Part Two: Collaborative Project Environment." A research project funded by the National Research Council's Industrial Research Assistance Program.

17. Crawford, M., Cann, J., and O'Leary, R. (1997). Uniclass: Unified Classification for the Construction Industry. RIBA Publications, London, UK.

18. Darwiche, A., Levitt, R., and Hayes-Roth, B. (1988). "OARPLAN: Generating Project Plans in a Blackboard System by Reasoning about Objects, Actions, and Resources." Artificial Intelligence in Engineering Design, Analysis, and Manufacturing, Vol. 2, No. 3.

19. Dzeng, R. and Tommelein, I. (1995). "Case-based Scheduling Using Product Models." Proceedings of the 2nd Congress on Computing in Civil Engineering, Atlanta, GA, ASCE, pp. 163-170.

20. Dzeng, R. and Tommelein, I. (1997). "Boiler Erection Scheduling Using Product Models and Case-Based Reasoning." ASCE Construction Engineering and Management, Vol. 123, No. 3, pp.338-347.

21. Echeverry, D. (1991). "Factors for Generating Initial Construction Schedules." Thesis presented to the University of Illinois at Urbana-Campaign, in partial fulfillment of the requirements for a Ph.D. degree.

22. Echeverry, D., Ibbs, C. and Kim, S. (1990). "Sequencing Knowledge for Construction Scheduling." ASCE Construction Engineering and Management, Vol. 117, No. 1., pp. 118-130.

23. Fischer, M. and Aalami, F. (1996). "Scheduling with Computer-Interpretable Construction Method Models." ASCE Journal of Construction Engineering and Management, Vol. 122, No. 4, pp. 337-347.

24. Fischer, M., and Froese, T. (1996). "Examples and Characteristics of Shared Project Models." ASCE Computing in Civil Engineering, Vol. 10, No. 3, pp. 174-182.

25. Fischer, M., Luiten, G. and Aalami, F. (1995). "Representing Project Information and Construction Method Knowledge for Computer-Aided Construction Management." Computing in Civil Engineering: Proceedings of 2nd International Computing Congress, ASCE, pp. 404-414.

26. Flemming, U., Coyne, R., and Snyder, J. (1994). "Case-Based Design in the SEED System." Proc. 1st Congress on Computing in Civil Engineering, ASCE.

27. Ford, S. et al. (1994). "The Object Oriented Modeling of Building Design Concepts." Building and Environment, Vol. 29, No. 4, pp. 411-418.

28. Froese, T. (1992). "Integrated Computer-Aided Project Management Through Standard Object Oriented Models." Thesis presented to Stanford University, Stanford CA, in partial fulfillment of the requirements for a Ph.D. degree.

29. Froese, T. (1995). "Total-Project Computer Systems for Construction Management." Natural Science and Engineering Research Council of Canada (NSERC) individual research grant proposal.

30. Froese, T. (1996). "Models of Construction Process Information." ASCE Computing in Civil Engineering, Vol. 10, No. 3, pp. 183-193.

31. Froese, T., Fischer, M., Grobler, F., Ritzenthaler, J., Yu, K., Sutherland, S., Staub, S., Akinci, B., Akbas, R., Koo, B., Barron, A., and Kunz, J. (1999) "Industry Foundation Classes for Project Management- A Trial Implementation." Electronic Journal of Information Technology in Construction, editor B-C. Bjoerk., published November 1999 at http://itcon.org/1999/2/.

32. Froese, T., Grobler, F., Yu, K. (1998). "Development of Data Standards for Construction – An IAI Perspective." The Life Cycle of Construction IT Innovations-Technology Transfer from Research to Practice, Proceedings of the W78 conference, Stockholm, Sweden, June 1998, KTH, Stockholm, pp. 233-244.

33. Froese, T., and Rankin, J. (1998). "Representation of Construction Methods in Total Project Systems" Computing in Civil Engineering: Proceedings of the 5th International Computing Congress, ASCE, Boston, MA, October 19-21, pp. 383-394.

34. Froese, T., Rankin, J., and Waugh, L. (1997a). "Facility Management Turnover System Data Model, Version 1.0." A consulting report prepared for Zoser Systems Group, San Francisco, CA.

35. Froese, T., Rankin, J., and Yu, K. (1997b) "Project Management Application Models and Computer -Assisted Construction Planning in Total Project Systems," International Journal of Construction Information Technology, (5) 1, pp. 39-49.

36. Froese, T., Rankin, J., and Yu, K. (1997c). "An Approach to Total Project Systems," Computing in Civil Engineering: Proceedings of the 4th International Computing Congress, ASCE, Philadelphia, PA, June, pp. 1-8.

37. Froese, T. and Russell, A. (1995). "Computer-Integrated Management Systems for Medium-Sized Contractors." Proceedings of Canadian Society of Civil Engineers Annual Conference, Ottawa, Ont., pp. 185-194.

38. Froese, T. and Yu, K. (1999). "Industry Foundation Class Modeling for Estimating and Scheduling." Proceedings of CIB 8th International Conference on Durability of Building Materials and Components, W78 Workshop, Vancouver, BC. Institute for Research in Construction, Ottawa, pp.2825-2835.

39. Gielingh, W. (1988). "General AEC Reference Model (GARM)." ISO TC 184/SC4/WG1 Document 3.2.2.1, TNO report BI-88-150, Building and Construction Research.

40. Gillespie, P. (1998). "A Rigorous Approach for Representing and Evaluating Construction Information Flow." Master of Science in Engineering Thesis, University of New Brunswick.

41. Gorlick, A. and Froese, T. (1999) "A Prototype Distributed CIC System Based on IAI Standards." Proceedings of CIB 8th International Conference on Durability of Building Materials and Components, W78 Workshop, Vancouver, BC. Institute for Research in Construction, Ottawa, pp.2825-2835.

42. Haley Enterprises (1997). Easy Reasoner Software Documentation, The Haley Enterprises, Sewickley, PA.

43. Hannus, M. et al. (1996). "STAR Research Program." a Word Wide Web document at http://www.vtt.fi.cic/projects/star/star.html, first visit December, 1996, last visited May, 1999.

44. Hannus, M. and Pietilainen, K. (1995). "Implementation Concerns of Process Modeling Tools." Computing in Civil Engineering: Proceedings of 2nd International Computing Congress, ASCE, pp. 449-458.

45. Hendrickson, C., Zozaya-Gorostiza, C., Rehak, D., Baracco-Miller, E., and Lin, P. (1987). "Expert System for Construction Planning." ASCE Computing in Civil Engineering, Vol. 1, No. 4, pp. 253-269.

46. IAI (1996). End User Guide to Industry Foundation Classes, Enabling Interoperability in the AEC/FM Industry. International Alliance for Interoperability (IAI), Washington, DC.

47. IAI (1997). Industry Foundation Classes 1.5. International Alliance for Interoperability (IAI), Washington, DC.

48. IAI (1998). Industry Foundation Classes 2.0. International Alliance for Interoperability (IAI), Washington, DC.

49. Ioannou, P.G., and Liu, L.Y. (1993). "Advanced Construction Technology System -ACTS." ASCE Construction Engineering and Management, Vol. 119, No.2, pp 288-306.

50. ISO (1992). "STEP Part 1: Overview and Fundamental Principles," ISO TC184/SC4/WGPMA.

51. ISO/TR 14177 (1994). "Classification of Information in the Construction Industry." Technical Report by Technical Committee ISO/TC 59, Building construction, Subcommittee SC13, Organization of information in the processes of design, manufacture and construction.

52. ISO (1995a). "The Nature of Industrial Data - Some Architectural Aspects and Issues for SC4." ISO TC184/SC4/WG10 Document N31.

53. ISO (1995b). "Industrial Automation Systems and Integration - Product Data Representation and Exchange - Building Construction Core Model." ISO WD 10303-106, version Str411.

54. ISO (1996). "Building Construction Core Model - Proposed ISO 10300 Part 106," Building Construction Core Model, versions T100 and T200, world wide web documents located at http://helios.bre.co.uk/ccit/info/ceic.htm, first visit December, 1996, last visited May, 1999.

55. ISO/TC59/SC13, (1996a). "Building Construction: Organization of Information about Construction Works Framework for Classification of Information." Draft ISO Standard, Geneva.

56. Jägbeck, A. (1994). "MDA Planner: Interactive Planning Tool Using Product Models and Construction Methods." ASCE Journal of Computing in Civil Engineering, Vol. 8, No. 4, pp. 536-554.

57. Jägbeck, A. (1998). "IT Support for Construction Planning – A System Design Based on Integrated Information." Doctoral Thesis, Royal Institute Of Technology, Construction and Economics, Stockholm.

58. Kartam, N. and Levitt, R. (1990). "Intelligent Planning of Construction Projects." ASCE Computing in Civil Engineering, Vol. 4, No. 2, pp. 155-176.

59. Kartam, N., Levitt, R., and Wilkins, D. (1991). "Extending Artificial Intelligence Techniques for Heirarchical Planning." ASCE Computing in Civil Engineering, Vol. 5, No.4, pp.464-477.

60. Koike, H. and Yoshihara, H, (1993). "Fractal Approaches for Visualizing Huge Hierarchies." Proceedings of the IEEE Symposium on Visual Languages, pp.55-60.

61. Lamping, J., Rao, R., and Pirolli, P. (1995). "A Focus+Content Technique Based on Hyperbolic Geometry for Visualizing Large Hierarchies." Proceedings of ACM Conference on Computer-Human Interaction, pp. 401-408.

62. Levitt, R., Kartam, N., and Kunz, J. (1988). "Artificial Intelligence Techniques for Generating Construction Project Plans." ASCE Construction Engineering and Management, Vol. 114, No. 3, pp.329-343.

63. Maher, M. and Balachandran, B. (1994) "Multimedia Approach to Case-Based Structural Design." ASCE Journal of Computing in Civil Engineering, 8(3), pp. 359-376.

64. Maher, M. and Gomez de Silva Garza, A. (1996) "Developing Case-Based Reasoning for Structural Design" IEEE Expert, 11(3) June, pp. 42-52.

65. Maher, M. and Gomez de Silva Garza, A. (1997). "Case-Based Reasoning in Design" IEEE Expert, AI in Design, March-April, pp. 34-41.

66. Microsoft (1997a). MS Access 97 Software Documentation, Microsoft Corp, Seattle WA.

67. Microsoft (1997b). MS Project 98 Software Documentation, Microsoft Corp, Seattle WA.

68. Microsoft (1998). MS Visual Basic 6.0 Software Documentation, Microsoft Corp, Seattle, WA.

69. National Institute of Standards and Technology (NIST) (1993a). Integration Definition for Function Modeling (IDEF0). Federal Information Processing Standards Publication 183, Computer Systems Laboratory, Gaithersburg, MD.

70. National Institute of Standards and Technology (NIST) (1993b). Integration Definition for Information Modeling Extended (IDEF1X). Federal Information

Processing Standards Publication 183, Computer Systems Laboratory, Gaithersburg, MD.

71. Navinchandra, D., Sriram, D., and Logcher, R. (1988). "GHOST: Project Network Generator." ASCE Computing in Civil Engineering, Vol. 2, No. 3, pp. 239-254.

72. Ng, S., Smith, N. and Skitmore, R. (1998). "A Case-Based Reasoning Model for Contractor Pre-qualification" International Journal of Construction Information Technology, 6(1), pp. 47-61.

73. NVision (1999). "Product Information Brief: NestedVision 3D." NVision Software Systems Inc.

74. O'Brien, M. (1997). "Integration at the Limit: Construction Systems." International Journal of Construction Information Technology, 5 (1), pp.89-98.

75. PMI (1996). A Guide to the Project Management Body of Knowledge. Project Management Institute (PMI), PMI Standards Committee, Upper Darby, PA.

76. Primavera Systems (1997). Sure-Trak 4.0, product information at http://www.primavera.com, first visited September 1997, last visited November 1999.

77. Rackley, S., Waugh, L, Froese, T. and Rankin, J. (1998). "Classification Hierarchies: A Proposed Graphical Interface." Proceedings of the CSCE Annual Conference, Halifax, NS, June 10-13. CSCE Montreal Canada. Vol. 1, pp.109-118

78. Rankin, J. and Froese, T. (1997). "Computer-Assisted Construction Planning (CACP) in the Context of Total Project Systems (TOPS)." Proceedings of the CSCE Annual Conference and 2nd Construction Specialty Conference, Sherbrooke, Quebec, pp. 41-50.

79. Rankin, J., Froese, T., and Waugh, L. (1998). "The Functionality of Computer-Assisted Construction Planning." Proceedings of the CSCE Annual Conference, Halifax, NS, June 10-13, pp. 119-128.

80. Rankin, J., Froese, T. and Waugh, L. (1999) "Exploring the Application of Case-Based Reasoning to Computer-Assisted Construction Planning." Proceedings of CIB 8th International Conference on Durability of Building Materials and Components, W78 Workshop, Vancouver, BC. Institute for Research in Construction, Ottawa, pp.2526-2536.

81. Ray-Jones, A. and Clegg, D. (1976) Construction Indexing Manual. SfB Agency, UK.

82. Reschke, R. and Teijgeler, H. (1994). "Generic Reference Model for Life Cycle Facility Management (a proposal)." ISO TC184/SC4/WG3 Document N351, Rev. 1.

83. Rezgui, Y., et al. (1996). "An Integrated Framework for Evolving Construction Models." The International Journal of Construction Information Technology, (4) 1, 47-60.

84. Riesbeck, C.K. and Schank, R.C. (1989). Inside Case-Based Reasoning. Lawrence Erlbaum Associates, Hillsdale, NJ.

85. Russell, A., and Wong, W. (1993). "New Generation of Planning Structures." ASCE Journal of Construction Engineering and Management, Volume 119, No. 2, pp. 196-214.

86. Russell, A., and Chevallier, N., (1998). "Representing a Project's Physical View in Support of Project Management Functions." Canadian Journal of Civil Engineering, Vol. 25, No. 4, pp. 707-717.

87. Sakar, M. and Brown, M. (1992). "Graphical Fisheye Views of Graphs." Digital Systems Research Center Research Report 084a.

88. Sanvido, V. (1990). "An Integrated Building Process Model." Computer Integrated Construction Research Program, Pennsylvannia Sate University, Technical Report No.1.

89. Schmitt, G. (1993) "Case-Based Reasoning in an Integrated Design and Construction System" International Journal of Construction Information Technology, 1(3), pp. 31-51.

90. Scitor (1997). Project Scheduler 7.0, product information at http://www.scitor.com, first visited September 1997, last visited November 1999.

91. Shaked, O. and Warszawski, A. (1995). "Knowledge-Based System for Construction Planning of High-Rise Buildings." ASCE Journal of Construction Engineering and Management, Vol. 121, No. 2, pp.172-182.

92. SHL System House (1997). Transform 7.0, evaluation software, product information at http://www.shl.com, first visited September 1997, last visited November 1999.

93. Stumpf, A., Chin, S., and Liu, L. (1995). "Use of a Relational Database System to Integrate Product and Process Information During Construction." CIB Proceedings W78, CIB Workshop on Computers and Information in Construction, Stanford University, pp.316-326.

94. Stumpf, A., Ganeshan, R., Chin, S., and Liu, L. (1996). "Object-Oriented Model for Integrating Construction Product and Process Information." ASCE Computing in Civil Engineering, Vol. 10, No. 3, pp. 204-212.

95. Sycara, K., Navinchandra, D., Guttal, R., Koning, J., and Narasimhan, S. (1992). "CADET: A Case-Based Synthesis Tool for Engineering Design" International Journal of Expert Systems, 4(2), pp. 157-188.

96. Tah, J., Carr V., Howes, R. (1998) "An Application of Cased-Based Reasoning to the Planning of Highway Bridge Construction" Engineering, Construction and Architectural Management, 5(4), pp. 327-338.

97. Tatum, C. (1989). "Classification System for Construction Technology" ASCE Construction Engineering and Management, Vol. 114, No.3., pp. 344-363.

98. Teicholz, P. and Fischer, M. (1994). "Strategy for Computer Integrated Construction Technology." ASCE Construction Engineering and Management, Vol. 120, No. 1, pp. 117-131.

99. Tracey, A., et al. (1996). "Developing Integrated Applications for Construction: The OSCON Approach." First International Conference on Computing & Information Technology for A/E/C, Singapore, pp. 361-368.

100. van Wezel, W., Jorna, R.J., and Mietus, D. (1996) "Scheduling in a Generic Perspective: Knowledge-Based Decision Support by Domain Analysis and Cognitive Task Analysis" International Journal of Expert Systems, 9(3), pp. 357-381.

101. Visio (1997). Visio Professional Software Documentation, Visio Corp, Seattle WA.

102. Voß, A. (1994). "Case-Based Reasoning in Building Design: Problems of Case Elicitation and Retrieval" International Journal of Construction Information Technology, 2(4), pp. 49-62.

103. Warszawski, A. and Sacks, R. (1995). "The Project Model of an Automated Building System." Computing in Civil Engineering: Proceedings of Second Congress ASCE, pp.77-89.

104. Warszawski, A. and Shaked, O. (1994). "Intelligent Construction Planning," ASCE Conference on Computing in Civil Engineering.

105. Watson, I. (1997). Applying Case-Based Reasoning: Techniques for Enterprise Systems, Morgan-Kaufmann Publishers, Inc., San Francisco, CA

106. Watson, I. (1998). "Is CBR a Technology or a Methodology?" Tasks & Methods in Applied Artificial Intelliegence. del Pobil., A.P., Mira, J. & Ali, M. (Eds.), pp. 525-534. Springer-Verlag Lecture Notes in Artificial Intelligence 1416, Berlin.

107. Waugh, L.M. (1990). "A Construction Planner." Thesis presented to Stanford University, Stanford CA, in partial fulfillment of the requirements for a Ph.D. degree.

108. Waugh, L.M. (1999). "Visualization of Hierarchical Construction Information." Natural Science and Engineering Research Council of Canada (NSERC) individual research grant proposal.

109. Winstanley, G., Chacon, M.A., and Levitt, R.E. (1993). "Model-Based Planning: Scaled-Up Construction Application." ASCE Computing in Civil Engineering, Vol. 7, No. 2, pp. 199-217.

110. Xpert Corporation (1997). PlanXpert, evaluation software, product information at http://www.planxpert.com, first visited September 1997, last visited November 1999.

111. Yamazaki, Y. (1993). "ICPS: Integrated Construction Planning System by Object-Oriented Planning Models." Computing in Civil Engineering: Proceedings of First Congress ASCE, pp.295-310.

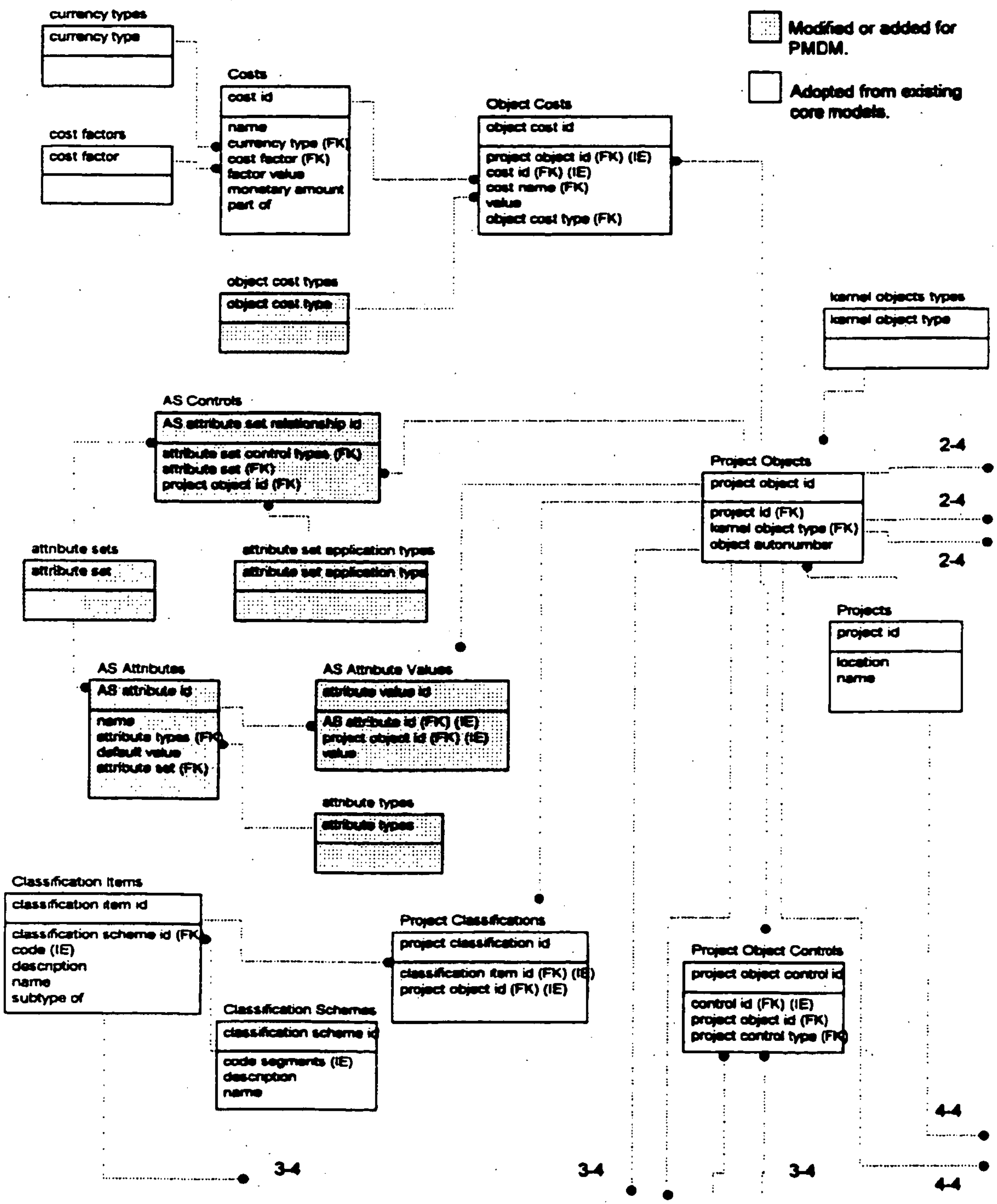
112. Yamazaki, Y. (1995). "An Integrated Construction Planning System using Object-Oriented Product and Process Modeling." Construction Management and Economics, Vol. 13, No. 1, pp. 417-426.

113. Yang, S-A and Robertson, D. (1995). "A Case-Based Reasoning System to Support the Relaxation of Building Regulations" International Journal of Construction Information Technology, 3(2), pp. 29-48.

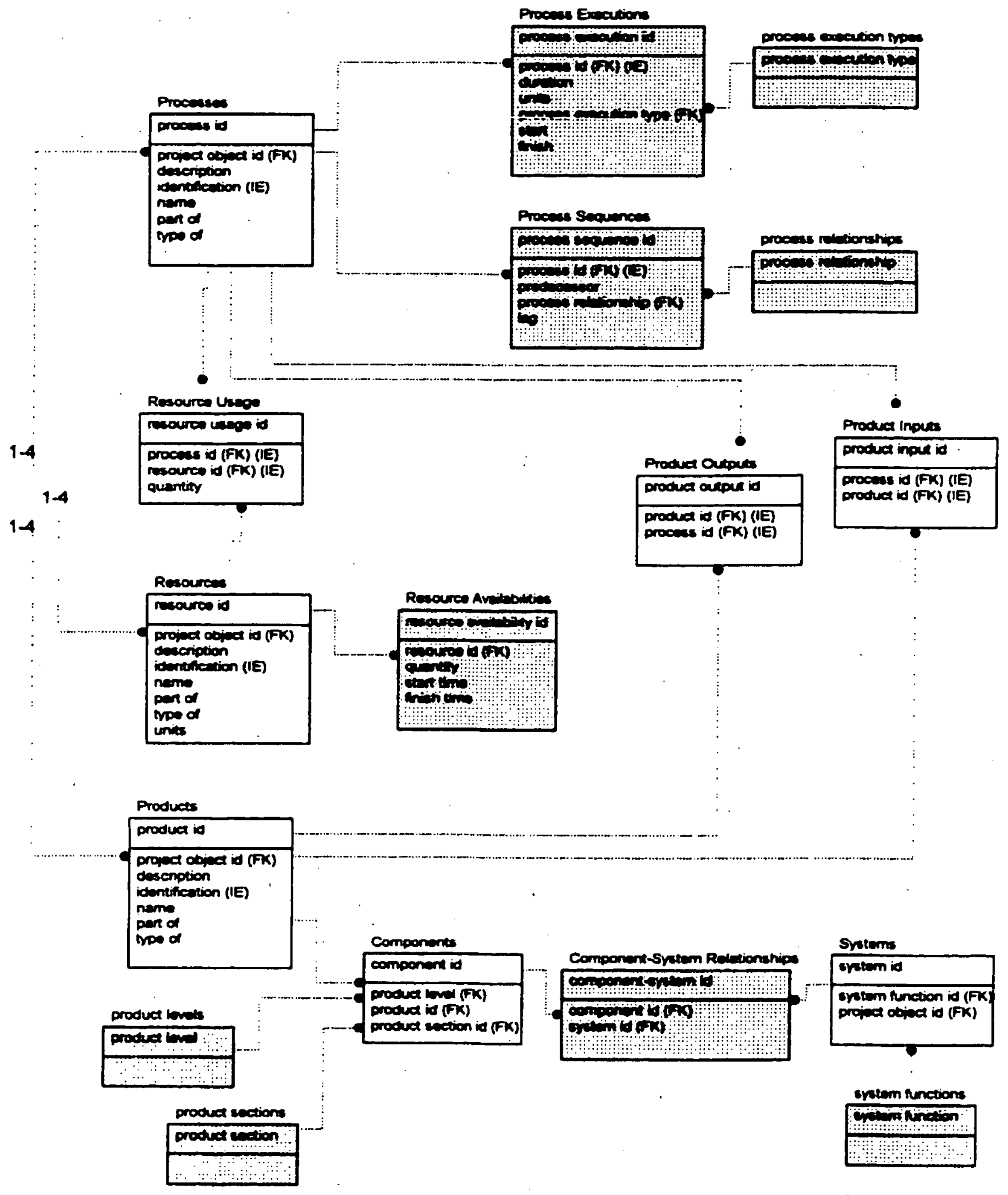
114. Yu, K. (1995). "Application of Project Information Models to Computer Integrated Construction." Master of Applied Science thesis, University of British Columbia, Vancouver, BC.

115. Zozaya-Gorostiza, C., Hendrickson, C., and Rehak, D. (1989). Knowledge-Based Process Planning for Construction and Manufacturing. Academic Press Inc., San Diego.

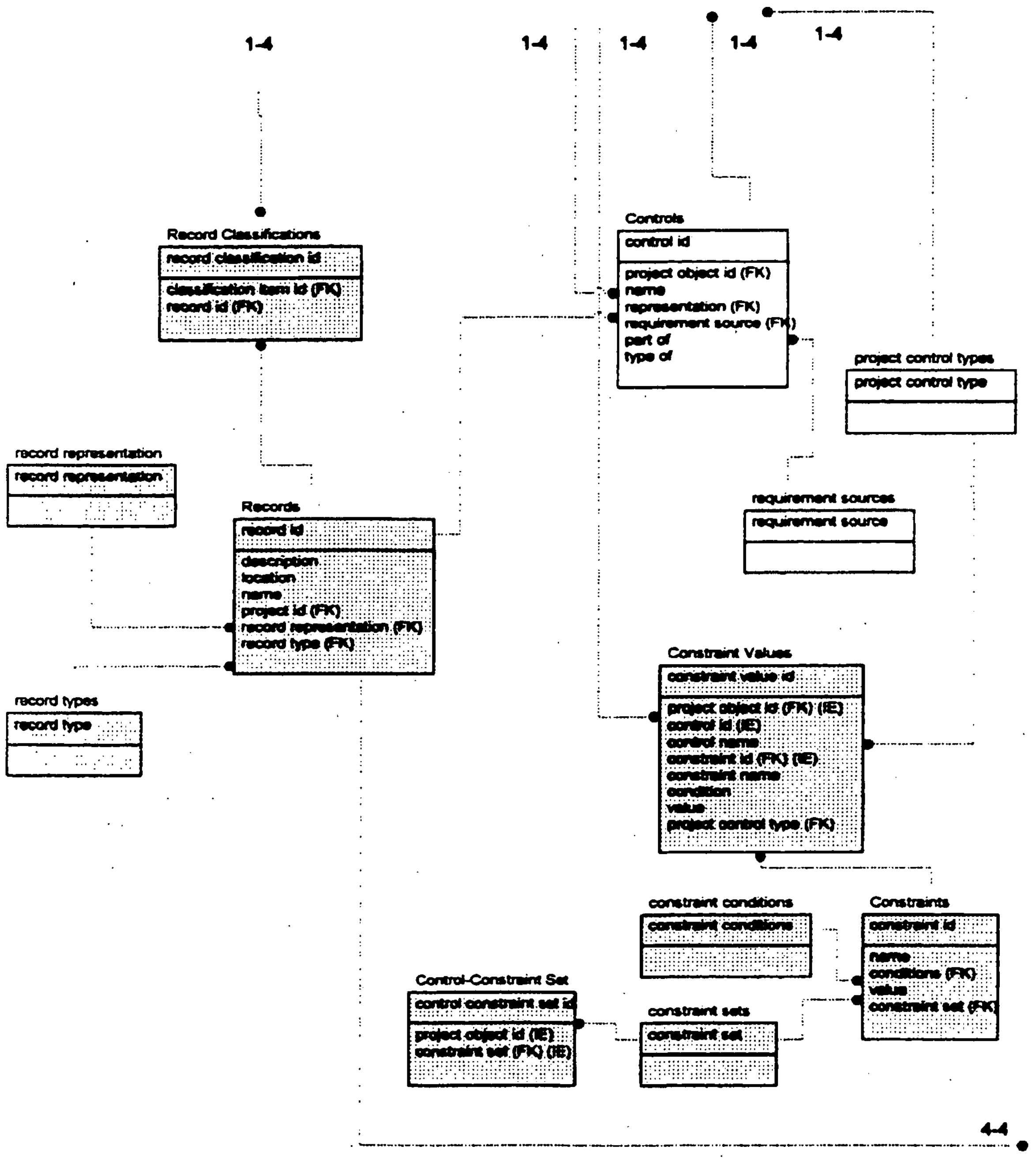
Appendix A: Consolidated PMDM Diagram, as Implemented



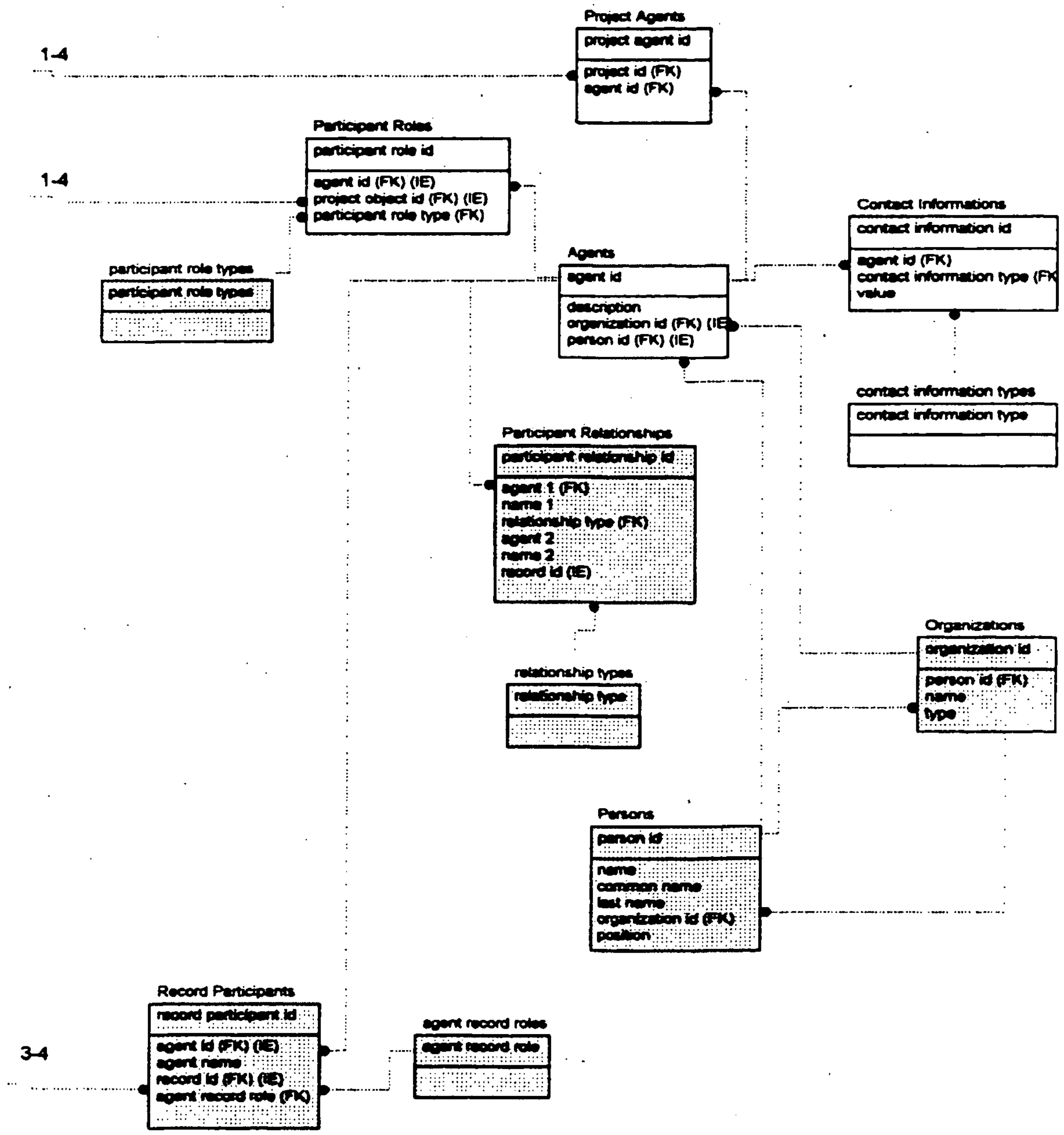
Page 1 of 4


Process Executions
process execution id
process id (FK) (IE)
duration
units
process execution type (FK)
start
finish
process execution types
process execution type
Processes
process id
project object id (FK)
description
identification (IE)
name
part of
type of
Process Sequences
process sequence id
process id (FK) (IE)
predecessor
process relationship (FK)
lag
process relationships
process relationship
Resource Usage
resource usage id
process id (FK) (IE)
resource id (FK) (IE)
quantity
1-4
1-4
1-4
Product Inputs
product input id
process id (FK) (IE)
product id (FK) (IE)
Product Outputs
product output id
product id (FK) (IE)
process id (FK) (IE)
Resources
resource id
project object id (FK)
description
identification (IE)
name
part of
type of
units
Resource Availabilities
resource availability id
resource id (FK)
quantity
start time
finish time
Products
product id
project object id (FK)
description
identification (IE)
name
part of
type of
Components
component id
product level (FK)
product id (FK)
product section id (FK)
Component-System Relationships
component-system id
component id (FK)
system id (FK)
Systems
system id
system function id (FK)
project object id (FK)
product levels
product level
product sections
product section
system functions
system function

Page 2 of 4


1-4
1-4
1-4
1-4
1-4
Record Classifications
record classification id
classification item id (FK)
record id (FK)
Controls
control id
project object id (FK)
name
representation (FK)
requirement source (FK)
part of
type of
project control types
project control type
record representation
record representation
Records
record id
description
location
name
project id (FK)
record representation (FK)
record type (FK)
requirement sources
requirement source
record types
record type
Constraint Values
constraint value id
project object id (FK) (IE)
control id (IE)
control name
constraint id (FK) (IE)
constraint name
condition
value
project control type (FK)
constraint conditions
constraint conditions
Constraints
constraint id
name
conditions (FK)
value
constraint set (FK)
Control-Constraint Set
control constraint set id
project object id (IE)
constraint set (FK) (IE)
constraint sets
constraint set
4-4

Page 3 of 4


1-4
1-4
3-4
Project Agents
project agent id
project id (FK)
agent id (FK)
Participant Roles
participant role id
agent id (FK) (IE)
project object id (FK) (IE)
participant role type (FK)
participant role types
participant role types
Agents
agent id
description
organization id (FK) (IE)
person id (FK) (IE)
Contact Informations
contact information id
agent id (FK)
contact information type (FK)
value
contact information types
contact information type
Participant Relationships
participant relationship id
agent 1 (FK)
name 1
relationship type (FK)
agent 2
name 2
record id (IE)
relationship types
relationship type
Organizations
organization id
person id (FK)
name
type
Persons
person id
name
common name
last name
organization id (FK)
position
Record Participants
record participant id
agent id (FK) (IE)
agent name
record id (FK) (IE)
agent record role (FK)
agent record roles
agent record role

Page 4 of 4

Appendix B: CACP Scope of Implementation

Implemented Visual Basic procedures for adding project information:

Public Sub AddAgents()	Adds a node as a child to the treeview, the parent is the focus node then adds to the database table.
Public Sub AddAttributeSets()	Adds an attribute set to project object with required relationships.
Public Sub AddClassItems()	Adds items to a classification scheme.
Public Sub AddControls()	Adds a node as a child to the treeview, parent is focus node then adds to the database also imposes a 1:1 relationship for controls:records.
Public Sub AddCosts()	Adds a node as a child to the treeview, the parent is the focus node then adds to the database.
Public Sub AddLibObjects()	Adds a node as a child to the treeview, the parent is the focus node then adds to the database.
Public Sub AddObjects()	Adds a node as a child to the treeview 1, the parent is the focus node then adds to the database.
Public Sub AddObjectsCheck()	Condition check before adding new objects.
Public Sub AddRelationships()	Adding the selected relationship from treeview 2.

Implemented Visual Basic procedures for deleting project information:

Public Sub DeleteAgents()	Deletes the focus agent from the treeview and database.
Public Sub DeleteAllChildren()	Deletes the focus object and children from the treeview and database.
Public Sub DeleteClassItems()	Deletes the focus classification from the treeview and database.
Public Sub DeleteControls()	Deletes the focus control from the treeview and database, and deletes corresponding record.

Public Sub DeleteCosts()	Deletes the focus cost from the treeview and database, and deletes corresponding records i.e., Object Costs.
Public Sub DeleteLibObjects()	Deletes the focus object from the treeview and database.
Public Sub DeleteObjectsCheck()	Condition checks before deleting.
Public Sub DeleteObjects()	Deletes the focus kernel object from the treeview and database.
Public Sub DeleteRelationships()	Deletes relationships between kernels objects.

Implemented general navigation Visual Basic procedures:

Public Sub ButtonControl()	Enables and disables toolbar buttons depending on treeview focus.
Public Sub CheckOpenDataBase()	Ensures a database has been opened.
Public Sub CollapseAllNodes()	Collapse all branches of the treeview.
Public Sub ExpandAllNodes()	Expands all branches of the treeview.
Public Sub FilterAttributeSets()	Filters the attributes to show based on the selected set.
Public Sub FilterConstraintSets()	Filters the constraints to show based on the selected set.
Public Sub FilteredTree()	Handles the display of multiple object types in the same tree.
Public Sub NodeClick()	Accesses details of nodes.
Public Sub NavigationControlOff()	Disables buttons and menu items.
Public Sub NavigationControlOn()	Turns on controls buttons and menu items depending on form focus.
Public Sub NewKnowledgebase()	Creates new project database.
Public Sub OpenTreeBrowser()	Open a new Browser for exploring objects.
Public Sub PasteTreeBranch()	Pastes previous cut or copied branch as child of focus node.
Public Sub PasteType()	Creates a new instance based on a copied object type.
Public Sub SelectKnowledgebase()	Allows user to selects a database from MS Access.
Public Sub SetAbstract()	Toggles abstract button.
Public Sub SetPartof()	Toggles part of button.

Public Sub SetType()	Sets type of value for project object to the value selected in a library.
Public Sub SetTypeof()	Toggles the type of button.
Public Sub ToggleAddRelations()	Toggles the treeview relationships by changing the form size and hiding controls.
Public Sub ToggleDetails()	Toggles the treeview details, by changing the form size.
Public Sub ToggleShowRelations()	Toggles the treeview relationships by changing the form size and hiding controls.
Public Sub UpdateTreeBrowser()	Updates the tree browser with current values.

Implemented Visual Basic procedures for displaying information in forms:

Public Sub DisplayAddRelations()	Displays treeview 2 for adding relationships between kernel objects.
Public Sub DisplayAgents()	Builds tree to display project agents.
Public Sub DisplayAgentRelations()	Displays agent interrelationships.
Public Sub DisplayAttributeSets()	Displays attributes in the tab view based on sets and project object selected.
Public Sub DisplayLibrary()	Displays hierarchy of selected library.
Public Sub DisplayClassHierarchy()	Displays hierarchy of classification items.
Public Sub DisplayClassifications()	Changes the text boxes for classifications.
Public Sub DisplayConstraints()	Displays constraints in the control's tab view.
Public Sub DisplayConstraintValues()	Displays constraints for each project object and its value in an editable database grid.
Public Sub DisplayContactInformation()	Displays contact info for a given agent.
Public Sub DisplayContextView()	Roots a tree at selected object and displays the new browser with selected context.
Public Sub DisplayControls()	Displays the controls in the tab view.
Public Sub DisplayCosts()	Displays the costs in the tab view.
Public Sub DisplayCostInfo()	Builds a tree to display project costs.
Public Sub DisplayCostSets()	Displays costs in the tab view based on sets and project object selected.
Public Sub DisplayGeneral()	Changes the textboxes for classifications.
Public Sub DisplayHierarchy()	Displays a hierarchy of project information.

Public Sub DisplayKernelSpecific()	Filling kernel specific detail tab.
Public Sub DisplayParticipants()	Displays the participants in the tab view.
Public Sub DisplayRecords()	Build tree to display controls & records.
Public Sub DisplayRelationHierarchy()	Binds the relationship objects for the class of objects selected.
Public Sub DisplayRelationships()	Creates a relationship hierarchy in treeview 3.
Public Sub DisplayRecordParticipants()	Filters record participants in a database grid.

Implemented Visual Basic procedures for utility functions:

Public Sub CopyType()	Copies a type from library for paste as an instance in the browser tree.
Public Sub CopyTreeBranch()	Copies branch objects to a temp table.
Public Sub CutTreeBranch()	Cuts from the selected node for paste or deletion.
Public Sub ExportTreeBranch()	Same as copy branch while setting a name for current form.
Public Sub GetAllChildren()	Gets all children of a selected object and add to a temp table.
Public Sub GetChildren(String)	Get immediate children of project object.
Public Sub GetParents(String)	Finds all objects between selected object and root and adds them to a temp table.
Public Sub GetSiblings(String)	Gets siblings of project object by getting children of parent node.
Public Sub ImportTreeBranch()	Same as paste tree branch but draws from export for temp tables.

Implemented case based reasoning procedures:

Public Sub AddCaseControls()	Gets selected method controls, adds to project.
Public Sub AddCaseProcesses()	Gets selected method processes, adds to project.
Public Sub AddCaseResources()	Gets selected method resources, adds to project.
Public Sub CBRBuildCases()	Builds a single MS Access table from relational tables.
Public Sub CBRBuilddBaseFile()	Builds a .dbf (dBaseIV) file based on the CBR Cases table.

Public Sub CBRBuildIndex()	Builds the CBR index from the .dbf file.
Public Sub CBRDisplayRetrieval()	Populates a list box on retrieval by filtering the data source.
Public Sub CBRGetMatch()	Compares presented case with .cbr file and returns distances and indexes from the case base.

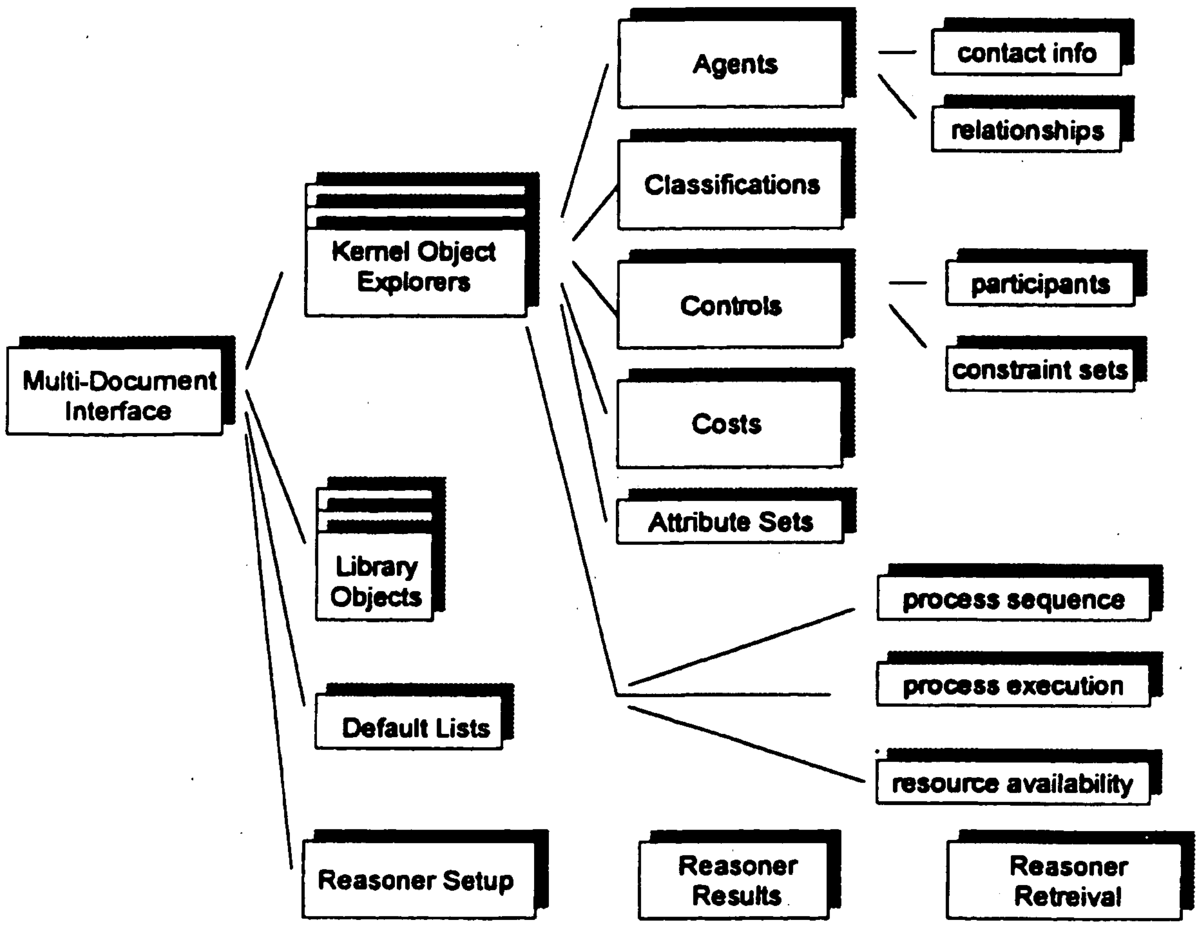


Figure B.1: Hierarchy of implemented Visual Basic forms.

Appendix C: Case-Based Reasoning Distance Calculations

The equations presented are based on The Haley Enterprise's "The Easy Reasoner" calculations for its "query by similarity" approach (The Haley Enterprise 1997). When calculating the distance between the target case and a potential solution from the case-base, each non-missing field value is used to calculate the distance along an axis in a multi-dimensional space where the number of dimensions equals the number of non-missing values. Each of these distances is normalized and can be weighted by the user; without weighting, each field contributes equally. The total distance will result in a value between 0 and 1 with a smaller distance indicating a closer match between the target case and potential solution. The equations for ordinal distances apply to fields with numerical data types (integers, dates, etc.) while nominal distance equations apply to textual fields.

Ordinal Distance

For ordinal distances the following equation is used to calculate a single field distance:

$$d_{ij} = \frac{| v_{ij} - V_j |}{| V_j \max - V_j \min |} \qquad [1]$$

where,
d_{ij} is the distance of a single ordinal dimension,
V_j is the target field value,
v_{ij} is the potential solution value,
$V_j$max is the maximum value of all potential solutions,
$V_j$min is the minimum value of all potential solutions.

Note: the minimum and maximum values are generated by the Easy Reasoner when the case-base contents are indexed.

Nominal Distance

To determine a nominal distance, a weight is calculated for each **term** in each text field of the case base and a **weight** is calculated for each term in the target to normalize the distance.

Text processing uses the values of the n-m gram to calculate the number of terms to consider for distance calculations. The n value is the number of non-blank characters in a term while the m value sets the step size.

For example, a 4-2 gram for the text field value *cast steel pruyn* yields the following six terms: cast, stst, stee, eelp, lpru, ruyn

Calculating the **weight** for each term uses the following equation:

$$W_{ik} = \frac{F_{ik} Log(n_k/N)}{\sqrt{\sum_j (F_{ij} Log(n_j/N))^2}} \qquad [2]$$

where,
F_{ik} is the number of times term k occurs in record i divided by the total terms in record i,

n_k is the number of different records in which term k occurs,
N is the number of records.

Note: the Easy Reasoner calculates the weights for each field's terms when the case base contents are indexed. Terms that are more frequently encountered in the case-base will receive a lower weight value than terms that are scarce (e.g., a term that is common to each record in the case-base has a weight of zero).

With calculated weights, the **similarity** (1-distance) between the target and a potential solution is calculated:

$$Si = \frac{\sum_{k=1}^{T} W_{ik} W_k}{\sqrt{\sum_{k=1}^{T} (W_{ik})^2 \sum_{k=1}^{T} (W_k)^2}} \qquad [3]$$

where,
W_k is the weight of the target field terms,
T is the number of terms in the target field.

Note: the closer the similarity value is to 1, the shorter the distance between the target case and the potential solution.

Total Distance

The total distance between the target case and a potential solution is calculated using the following equation:

$$Di = \frac{\sqrt{\sum_{j=1}^{N} d_{ij}^2}}{N} \qquad [4]$$

where,
N is the number of fields.

The user also has the ability to add **user-defined weights** (different from term weights) for each field included in the distance calculation. The total distance with user-defined weights is determined by:

$$Di = \frac{\sqrt{\sum_{j=1}^{N} (W_j d_{ij})^2}}{\sum_{j=1}^{N} W_j} \qquad [5]$$

where,

W is a user defined weight.

Example

A simple example relevant to CACP is considered in which the case base is comprised of five (5) potential solutions. The table below represents values for the target case presented and a potential solution for four (4) fields. Example values are provided for two (2) nominal and ordinal fields under the assumption that the *product type* and *process type* are the same for each case.

field type	**field**	**target value**	**case 1 value**
nominal constraint	pile point	cast steel pruyn	MAFCO cast steel
ordinal constraint	minimum length (ft)	20	25
nominal attribute	designation	HP 12x117	HP 12x74
ordinal attribute	section area (si)	34.0	21.7

Again, The Easy Reasoner calculates the required maximum and minimum field values for ordinal fields, and term weights for nominal fields when the case-base is indexed.

Applying equation [1] to the ordinal fields results in distance values of d_2=0.111 and d_4=0.427, while applying equation [2] and [3] to the nominal fields results in values of d_1=0.439 and d_3=0.1829.

Without considering user-defined weights for these fields, the total distance is calculated using equation [4] and gives a result of:

$$Di = \frac{\sqrt{((0.439)^2 + (0.111)^2 + (0.183)^2 + (0.427)^2)}}{4} = 0.162$$